



NETWORK PRODUCTS

**TRANSACTION FACILITY
VERSION 1
REFERENCE MANUAL**

**CDC® OPERATING SYSTEM:
NOS 1**

ALPHABETICAL COMMAND SUMMARY

BEGIN Request	2-28	LIBTASK Control Statement	9-1
BLDABH Request	2-12	LOADCB Request	2-9
BTRAN Request	5-1	LOGIN Request	2-31
BWAITINP Request	2-26	LOGT Request	2-29
CALLRTN Request	2-4	NEWTRAN Request	2-6
CALLTSK Request	2-5		
CEASE Request	2-29		
CMDUMP Request	6-3	RECALAL Request	2-28
		RELSCB Request	2-24
DSDUMP Request	6-1	SEND Request	2-16
		SETCHT Request	2-10
EXTRACT Request	2-31	STRAN Macro	11-1
		SUBMT Request	5-3
GETABH Request	2-13	TARO Request	2-25
		TERMDEF Request	2-15
HIO Request	2-26	TIMCNT Macro	11-1
INSERT Request	2-31	TPSTAT Request	2-27
ITL Request	2-29	TRAN Macro	11-1
		TRANCHK Request	2-6
JOURNAL Request	4-5	TSIM Request	2-24
		VALNET Statement	8-1
K.DSDUMP Command	6-4	WAIT Request	2-30
K.DUMP Command	6-5	WAITINP Request	2-8
K.DUMPLIM Command	6-5		
KPOINT Request	2-30		
KTSDMP Request	6-5		



NETWORK PRODUCTS

**TRANSACTION FACILITY
VERSION 1
REFERENCE MANUAL**

**CDC® OPERATING SYSTEM:
NOS 1**

[illegible]

ii

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Front Cover	-	6-2	D	F-5	D				
Inside Front Cover	D	6-3	D	F-6	D				
Title Page	-	6-4	D	F-7	C				
ii	D	6-5	D	F-8	C				
iii/iv	D	6-6	D	G-1	C				
v	D	7-1	D	H-1	D				
vi	D	7-2	D	Index-1	D				
vii	D	8-1	D	Index-2	D				
viii	D	8-2	D	Index-3	D				
1-1	D	8-3	D	Comment Sheet	D				
1-2	D	9-1	D	Back Cover	-				
1-3	D	9-2	D						
1-4	D	9-3	D						
1-5	D	9-4	D						
1-6	D	9-5	D						
2-1	D	10-1	D						
2-2	D	10-2	D						
2-3	D	11-1	D						
2-4	D	11-2	D						
2-5	D	11-3	D						
2-6	D	11-4	D						
2-7	D	11-5	D						
2-8	D	A-1	C						
2-9	D	A-2	C						
2-10	D	B-1	C						
2-11	D	B-2	D						
2-12	D	B-3	D						
2-13	D	B-4	D						
2-14	D	B-5	D						
2-15	D	B-6	D						
2-16	D	B-7	D						
2-17	D	B-8	D						
2-18	D	B-9	D						
2-19	D	B-10	D						
2-20	D	B-11	D						
2-21	D	B-12	D						
2-22	D	B-13	D						
2-23	D	B-14	D						
2-24	D	B-15	D						
2-25	D	B-16	D						
2-26	D	B-17	D						
2-27	D	B-18	D						
2-28	D	B-19	D						
2-29	D	B-20	D						
2-30	D	B-21	D						
2-31	D	B-22	D						
2-32	D	C-1	D						
3-1	D	D-1	D						
4-1	D	D-2	D						
4-2	D	E-1	D						
4-3	D	E-2	D						
4-4	D	E-3	D						
4-5	D	E-4	D						
4-6	D	E-5	D						
5-1	D	E-6	D						
5-2	D	F-1	D						
5-3	D	F-2	C						
6-1	D	F-3	D						
		F-4	D						

PREFACE

The CONTROL DATA® Transaction Facility (TAF), version 1.2, is a network host product. It is a network application that requires the facilities of the Network Access Method (NAM) and the 255x Series Communications Control Program (CCP). TAF runs as a subsystem of the Network Operating System (NOS) version 1 and may be used with COMPASS, FORTRAN, or COBOL.

This manual documents the executive portion of TAF. The remaining portion, the data manager supplied with TAF, is documented in the TAF Data Manager Reference Manual. Three other data managers, the CDC® CYBER Record Manager, the CYBER Database Control System (CDCS), and the Total data manager, are also provided. Refer to the list of related publications.

AUDIENCE DEFINITION

This manual is for systems analysts and applications programmers. A systems analyst is a programmer who knows COMPASS and is concerned with internal design and functions. An applications programmer is a programmer writing tasks in FORTRAN or COBOL. New users of TAF should consult the Network Products Transaction Facility User's Guide as a first source of information. It is an introductory text for inexperienced users.

The user must be familiar with basic NOS operations such as running a compilation job. The NOS Batch User's Guide acquaints the new user with the fundamentals of NOS. The NOS Reference Manual, volume 1, is for the high-level language applications programmer who is already familiar with NOS. The NOS Reference Manual, volume 2, is for the COMPASS applications programmer.

TERMINOLOGY AND CONVENTIONS

References in this manual to the transaction subsystem imply the whole of TAF. References to the transaction executive imply the controlling portion of TAF, the portion described in this manual. References to FORTRAN imply either FORTRAN Extended version 4 or 5; TS mode is not allowed (that is, the SEQ, TS, and OPT=0 parameters are not allowed on the FTX control statement). References to COBOL imply either COBOL version 4 or 5. (The UC1 parameter must be included on the COBOL5 control statement.)

The term alphanumeric refers to any combination of the letters A through Z and the numbers 0 through 9; special characters are not included.

Conventions for word formats are as follows:

- Cross-hatching indicates a field is not used by or is not applicable to TAF.
- Reserved fields may contain information useful to TAF internal processing. The user should not expect them to contain any particular values.

- Fields labeled with mnemonics indicate that a specific parameter must be inserted (generally described after the word format).
- Fields with numeric identifiers indicate the actual value that is used or returned for a particular request.

Extended memory for the CYBER 170 Model 176 is large central memory extended (LCME). Extended memory for all other NOS computer systems is extended core storage (ECS) or extended semiconductor memory (ESM).

In this manual, the acronym ECS refers to all forms of extended memory unless otherwise noted.

Programming information for the various forms of extended memory can be found in the COMPASS Reference Manual and in the appropriate computer system hardware reference manual.

RELATED PUBLICATIONS

The following manuals contain additional information for the user.

<u>Control Data Publication</u>	<u>Publication Number</u>
COBOL Version 4 Reference Manual	60496800
COBOL Version 5 Reference Manual	60497100
Common Memory Manager Version 1 Reference Manual	60499200
COMPASS Version 3 Reference Manual	60492600
CYBER Database Control System Version 2 Reference Manual	60481800
CYBER Loader Version 1 Reference Manual	60429800
FORTRAN Extended Version 4 Reference Manual	60497800
FORTRAN Version 5 Reference Manual	60481300
Network Products Interactive Facility Version 1 Reference Manual	60455250
Network Products Interactive Facility Version 1 User's Guide	60455260

<u>Control Data Publication</u>	<u>Publication Number</u>
Network Products Network Access Method Version 1 Network Definition Language Reference Manual	60480000
Network Products Network Access Method Version 1 Reference Manual	60499500
Network Products Transaction Facility Version 1 CYBER Record Manager Data Manager Reference Manual	60456710
Network Products Transaction Facility Version 1 Data Manager Reference Manual	60455350
Network Products Transaction Facility Version 1 User's Guide	60455360
NOS Version 1 Batch User's Guide	60436300
NOS Version 1 Installation Handbook	60435700
NOS Version 1 Manual Abstracts	84000420
NOS Version 1 Operator's Guide	60435600
NOS Version 1 Reference Manual, Volume 1	60435400

<u>Control Data Publication</u>	<u>Publication Number</u>
NOS Version 1 Reference Manual, Volume 2	60445300
NOS Version 1 System Maintenance Reference Manual	60455380
Total-CDC Version 1 Reference Manual	76070300

The NOS Manual Abstracts is a pocket-sized manual containing brief descriptions of the contents and intended audience of all NOS and NOS product manuals. The abstracts can be useful in determining which manuals are of greatest interest to a particular user.

Control Data also publishes a Software Publications Release History, publication number 60481000 of all software manuals and revisions packets it has issued. This history lists the revision level of a particular manual that corresponds to the level of software installed at the site.

DISCLAIMER

This product is intended for use only as described in this document. Control Data cannot be responsible for the proper functioning of undescribed features or parameters.

CONTENTS

1. INTRODUCTION	1-1	TAF Data Manager xxJ File	4-1
On-Line Transaction Processing Advantages	1-1	CRM Data Manager xxJ File	4-2
On-Line Transaction Processing Problems	1-1	Total Data Manager xxJ File	4-3
Batch Interface	1-3	CDCS or No Data Manager xxJ File	4-3
Network Interface	1-3	Creating xxJ File	4-4
Tasks	1-3	Journal Files	4-4
Transaction Processing	1-3	System Journal File	4-4
TAF Requests	1-4	Data Base Journal Files	4-4
Entering Transaction Subsystem	1-4	Journal File Entry Header	4-4
Implementing an Application	1-5	JOURNL Request	4-5
Subcontrol Points	1-6	TAF Configuration File	4-6
Multimainframe	1-6		
Extended Core Storage (ECS)	1-6		
2. TASK/TRANSACTION EXECUTIVE INTERFACE	2-1	5. BATCH/TRANSACTION SUBSYSTEM INTERACTION	5-1
Communication Block	2-1	BTRAN Request	5-1
Requests	2-1	SUBMT Request	5-2
Data Manager Requests	2-3	6. MEMORY DUMPS	6-1
Journal File Requests	2-3	Task Dump	6-1
Memory Dump Requests	2-3	DSDUMP Request	6-1
System Requests	2-3	CMDUMP Request	6-3
Memory Requests	2-3	K.DSDUMP Command	6-4
Task Scheduling Requests	2-3	KTSDUMP Utility	6-5
CALLRTN Request	2-4	Memory Dump Formats	6-5
CALLTSK Request	2-5	TAF Dump	6-5
NEWTRAN Request	2-6	K.DUMP Command	6-5
TRANCHK Request	2-6	K.DUMPLIM Command	6-5
Input/Output Requests	2-7	7. RECOVERY/OPERATOR INTERFACE	7-1
WAITINP Request	2-8	Recovery	7-1
LOADCB Request	2-9	Operator Interface	7-2
SETCHT Request	2-10	8. NETWORK FILES	8-1
BLDABH Request	2-12	Terminal Network Description File	8-1
GETABH Request	2-13	VALNET Validation	8-1
TERMDEF Request	2-15	Network File Organization	8-2
SEND Request	2-16	9. TASK LIBRARY	9-1
RELSCB Request	2-24	LIBTASK Utility	9-1
TSIM Request	2-24	Task Residency and Type	9-1
TARO Request	2-25	LIBTASK Control Statement	9-2
IIO Request	2-26	Input Directives	9-2
BWAITINP Request	2-26	Scheduling Priority	9-4
TPSTAT Request	2-27	Creating a New Task Library	9-4
BEGIN Request	2-28	Updating an Existing Task Library	9-5
Task Control Requests	2-28	Deleting a Task from a Task Library	9-5
RECALAL Request	2-28	Ignoring a Task in the Replacement File	9-5
CEASE Request	2-29	10. REQUIRED SYSTEM TASKS	10-1
ITL Request	2-29	Initial Task (ITASK)	10-1
LOGT Request	2-29		
KPOINT Request	2-30		
WAIT Request	2-30		
Task Utility Requests	2-31		
LOGIN Request	2-31		
EXTRACT Request	2-31		
INSERT Request	2-32		
3. ACCESS AND USAGE OF CDCS BY TASKS	3-1		
4. SYSTEM FILES	4-1		
xxJ File	4-1		

KDIS	10-1	K.IDLE Console Operator Command	11-3
Log-Out Task (LOGT)	10-1	TAF Subsystem Recovery	11-3
Message Abort (MSABT)	10-1	Terminal Login	11-3
OFFTASK	10-1	Terminal Break Conditions	11-3
SYMSG	10-2	Terminal Connection Broken	11-4
Execute Named Task (XTASK)	10-2	Network Normal Shutdown	11-4
		Network Abort	11-4
		Network Immediate Shutdown	11-4
		Terminal Inactive	11-4
		Logical Error	11-4
		Block Not Delivered	11-4
		Terminal Characteristic Changes	11-4
		Terminal Input Too Large	11-5
		Stop and Start on Downline	
		Connection	11-5
		Terminal Type Ahead	11-5
		User Status	11-5
11. ITASK SYSTEM TASK	11-1		
Macros	11-1		
TIMCNT Macro	11-1		
TRAN Macro	11-1		
STRAN Macro	11-1		
ITASK Description	11-1		
TAF-Originated Transactions	11-2		
Conditions Requiring TAF-Originated			
Transactions	11-3		
Time Activation	11-3		

APPENDIXES

A. CHARACTER SETS	A-1	E. INSTALLATION OVERVIEW	E-1
B. ERROR MESSAGES	B-1	F. TERMINAL DEFINITION COMMANDS	F-1
C. SAMPLE DECK STRUCTURES	C-1	G. LINE FEED AND TRANSMISSION KEYS	G-1
D. FILE DESCRIPTIONS AND DEFINITIONS	D-1	H. DETECTION OF POTENTIALLY BLOCKED TASKS	H-1

INDEX

FIGURES

1-1 TAF Control Point for CRM, TOTAL, and TAF Data Manager	1-2	2-18 ITL Parameter Table	2-30
1-2 TAF Control Point for CDCS Data Base Manager	1-2	4-1 Journal File (Buffered and Nonbuffered Structure)	4-2
2-1 Communication Block	2-2	4-2 Journal File Entry Header	4-4
2-2 CALLRTN	2-4	4-3 JOURNAL Parameter Format	4-5
2-3 CALLRTN (Chain)	2-5	5-1 BTRAN Header Word	5-1
2-4 CALLRTN (Multiple Chains)	2-5	5-2 BTRAN Request	5-2
2-5 CALLTSK With CEASE	2-6	5-3 SUBMT Parameter Table	5-3
2-6 CALLTSK Without CEASE	2-6	5-4 SUBMT Request	5-4
2-7 NEWTRAN Parameter Table	2-7	6-1 DSDUMP Parameter Table	6-3
2-8 TRANCHK Parameter Table	2-8	6-2 CMDUMP Parameter Table	6-4
2-9 LOADCB Parameter Table	2-10	6-3 KTSDMP Format	6-6
2-10 SETCHT Parameter Table	2-11	8-1 Communication Between Terminal Status Table and Communication Block	8-2
2-11 GETABH Parameter Table	2-14	11-1 Format of TTOT Table Entry	11-2
2-12 TERMDEF Parameter Table	2-16	11-2 Format of TRANT Table Entry	11-2
2-13 SEND Parameter Table	2-22	11-3 Format of TRANS Table Entry	11-2
2-14 TSIM Parameter Table	2-25	11-4 Communication Block for TAF-Originated Transactions	11-4
2-15 TARO Parameter Table	2-27		
2-16 IIO Parameter Table	2-27		
2-17 TPSTAT Return Values	2-28		

TABLES

2-1 BLDABH Request Parameters	2-12	2-4 FORTRAN SEND Request Parameters	2-19
2-2 Application Block Header	2-14	2-5 COMPASS SEND Request Parameters	2-20
2-3 COBOL SEND Request Parameters	2-17	2-6 COMPASS SEND Parameter Interrelationships	2-23

TAF is a network product that controls transaction processing. Transaction processing consists of taking an existing collection of information, called a data base, and correcting old data or adding new data to create an up-to-date data base. Such a correction or addition is called a transaction.

In a transaction processing system, the data base must be efficiently structured and easily accessible to the user. A data manager provides these features.

Control Data offers the following data managers for use with TAF.

- The CYBER Database Control System (CDCS).
- The CYBER Record Manager (CRM). (TAF supports only those features documented in the TAF CRM Data Manager Reference Manual.)
- The Total data manager.
- The TAF Data Manager.

All four data managers can operate concurrently. However, a single transaction can use only:

- CDCS and one other data manager,
or
- CDCS alone,
or
- One other data manager alone.

The main function of TAF is on-line transaction processing (direct operator interface with the data base through a terminal).

ON-LINE TRANSACTION PROCESSING ADVANTAGES

Some advantages associated with on-line transaction processing are:

- The operator submitting the data through TAF can make immediate data corrections.
- The operator can verify that operations have completed successfully. TAF allows the operator to view additions or corrections to the data base to verify their correctness.
- The data base is updated in seconds. TAF provides immediate updating capability which allows other transactions to make use of the updated data base.

The most important feature of TAF is its efficient system resource utilization. TAF's transaction processing differs from other modes of operation in that a task only contains application code. Terminal communications are handled by one terminal communications interface. Data base communications are handled by one copy of a data base manager. When tasks need to be loaded, they are simply copied from a task library to a subcontrol point; none of the normal overhead of loading is incurred.

The TAF control point contains application tasks as subcontrol points. When a task completes, one of the following occurs:

- It may be immediately reused by other waiting transactions (no reloading is necessary).
- Another task may be loaded in its place.
- Other active tasks may be moved up in memory and the remaining memory returned to the system for use by non-TAF applications.

TAF will expand its memory to a maximum value as needed.

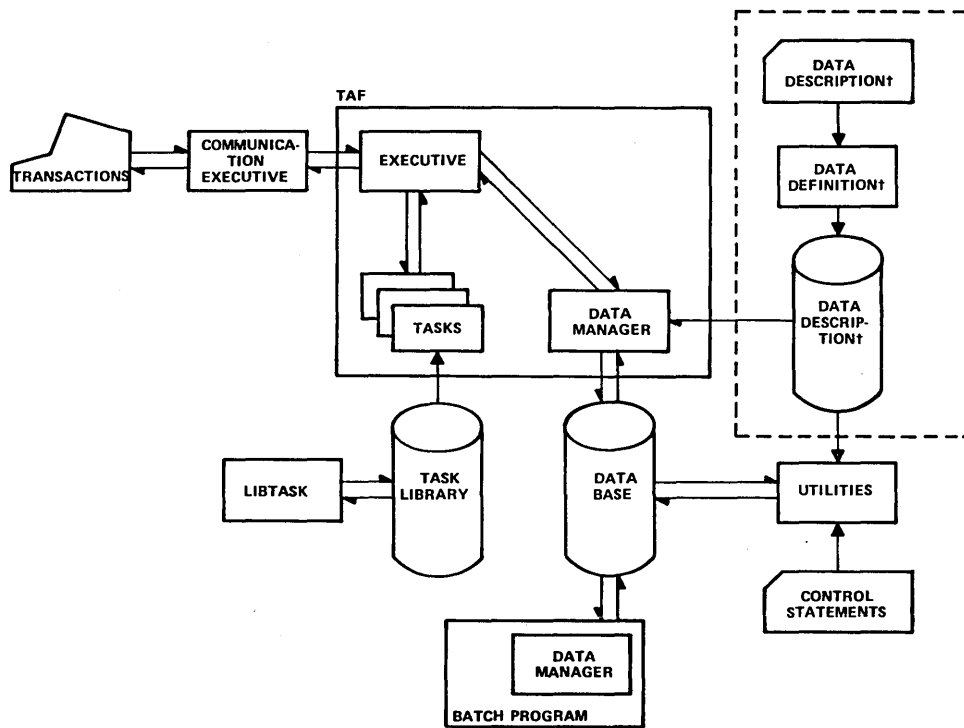
When there is no transaction activity, TAF size is reduced to a small amount of memory. When input is sensed, TAF rolls back in to service the request.

Figures 1-1 and 1-2 summarize the TAF control point and its operation.

ON-LINE TRANSACTION PROCESSING PROBLEMS

Some problems associated with on-line transaction processing and the methods that TAF uses to alleviate these problems are:

- It is difficult to schedule and control multiple jobs. TAF solves the problem of multiple job control scheduling by using subcontrol points. Like the operating system which controls and schedules jobs to control points (where they are processed), TAF controls and schedules transaction processing programs to portions of its control point called subcontrol points. Subcontrol points are protected from each other by CYBER hardware memory protection.
- The terminal interface is complicated. TAF minimizes terminal programming by using simple input/output (I/O) commands that are processed by the standard network terminal interface.



†COMPONENTS INSIDE DOTTED LINE NOT APPLICABLE TO CRM.

Figure 1-1. TAF Control Point for CRM, TOTAL, and TAF Data Manager

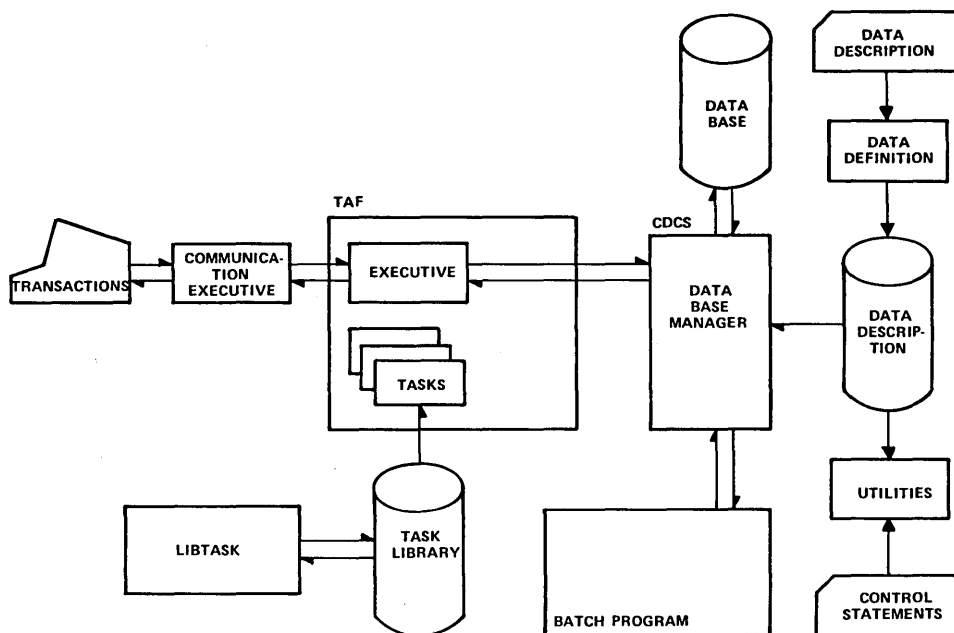


Figure 1-2. TAF Control Point for CDCS Data Base Manager

BATCH INTERFACE

A batch interface with TAF is also available. The batch interface can generate large reports without tying up a terminal until the listing has been printed. The report can be printed at a high-speed central site printer. Batch mode can also be used for testing and debugging. The batch interface is described in section 5.

NETWORK INTERFACE

The network terminal interface uses NAM and 255x communications processors. This interface is used by all other network products (for example, Remote Batch Facility and Interactive Facility). If validated, terminal users may select these applications as well as TAF. The network provides the following features for transaction processing.

- Managing network protocol.
- Buffering and queuing data for regulating data flow.
- Supporting a wide variety of terminals through standardization of data formats as well as providing the ability to handle transparent data.
- Sharing network among communication-oriented services.

The network makes use of the interactive virtual terminal (IVT) concept. This concept abstracts certain functions of a variety of real terminals. The application programmer need not be concerned with character sets and communications protocols.

- An IVT has an input/output (I/O) device that sends or receives certain amounts of data termed logical lines. These logical lines are transformed into physical lines of characters of the appropriate character set for the real terminal.
- Logical lines which exceed the physical capacity of the real terminal are automatically folded into two or more physical lines.
- The spacing of logical lines of output can be further controlled by the use of format effectors.
- Output to a device may be optionally paged, so that data which would overwrite any output being displayed is not sent until the terminal user has acknowledged the preceding output.
- Both the application and the terminal user can redefine terminal characteristics of the actual terminal, which may differ from those assumed, and supply or redefine the operational controls provided.

When an application requires features not provided by the IVT but known to exist on the connected real terminal, the application may do one of the following.

- Embed appropriate control characters in the output text.
- Transfer data in transparent mode, in which case, all transforms are inhibited, and the application has direct access to and responsibility for all real terminal features including the character set.

TASKS

The applications programmer implements a transaction system by writing tasks. A task is simply a program with a TAF interface that performs a specific function. A single task or several tasks perform a sequence of events called a transaction. Tasks can read and update information on the user's data base. They can send messages to terminals and receive input. Tasks can also schedule other tasks to assist in the completion of transactions.

Tasks that run under TAF can be written in COBOL, FORTRAN, or COMPASS. A task differs from a normal batch or time-sharing program in that it contains calls to TAF commands and data manager commands and must have a common block (referred to as the communication block) defined. The communication block is needed to receive terminal input and pass data between tasks. (The communication block is described in section 2.)

Binary copies of each task are stored in the task library. The LIBTASK utility is used to place these binary copies on the task library. Section 9 describes the task library and LIBTASK.

TRANSACTION PROCESSING

After following the log-in procedures outlined later in this section, the user indicates the transaction to be performed by entering a code that is assigned by the system designers. Such a code could specify that a user is entering a loan payment; another code would be used to close out a loan, and so on. TAF receives this initial input and places it in a communication block. This communication block is the means by which the data is passed from a terminal to a task or from task to task.

When a transaction is completed, the communication block is released and can be used by other transactions.

After the communication block has been constructed, TAF schedules an initial task (ITASK) to interpret the code the user entered. Using this code, ITASK requests TAF to schedule the appropriate task to process the transaction. The task scheduled may also call other tasks to assist in the completion of the transaction.

TAF REQUESTS

Communication between terminals and tasks is accomplished by using TAF requests. Output can be sent to a transaction terminal from the task by the use of the SEND request. When this request is processed, control passes to TAF. TAF extracts the message from the task and passes it to the network. The message is then transmitted to the transaction terminal, and control passes back to the task for additional processing.

When a task requires additional input from the terminal, the WAITNP request is used. The task performs a SEND request to send output to the terminal, prompting the terminal operator for the needed input. The WAITNP request is then processed. Since the terminal operator takes seconds to respond to the request, the task is rolled out and the subcontrol point and communication block are released. These resources can then be used to process other transactions while the terminal operator is responding. When TAF receives the requested input, it places it into a new communication block, and the task is rolled into an available subcontrol point. The new communication block is passed to the task, and the task continues processing.

These and other requests are further described in sections 2 and 5.

ENTERING TRANSACTION SUBSYSTEM

This section discusses various procedures which can be provided for accessing the transaction subsystem. The standard access procedure is described first. Also discussed are alternatives to this procedure and additional information to be supplied to the user. These areas are mentioned so that a coordinator of TAF activities at a particular site can determine, knowing the types of users and applications at the site, the most desirable form of access procedures for that site.

To access the transaction subsystem, the terminal must be recognized as part of the network and then be logged into the computer system. To do this, the following procedure is required.

1. Connect the terminal to the data line. For a hardwired terminal, this is already done; for a dial-up terminal, this means calling the computer center and connecting the phone to the terminal.
2. Identify the terminal to the network. This is not always required but may be necessary for the network to determine correctly the type and speed of the terminal. Terminal identification is performed in the following manner.

Press CR [†]

The user has 60 seconds to press CR . This determines the line speed.^{††} The network prompts the user with a line feed to indicate that the character set recognition part of terminal identification can be entered.^{†††}

Press) CR

The user has 60 seconds to press) CR . The network issues a line feed to indicate that terminal identification is finished.

NOTE

For standard ASCII terminals, the user presses the first CR . After receiving the line feed prompt, the user can press CR to bypass the character set recognition part of terminal identification.

The network identifies the terminal and establishes a data path so that the log-in sequence can begin.

When the network determines that a data path exists between the terminal and the computer system, the log-in sequence begins. This sequence may vary at some sites due to the flexibility of the network.

Prior to initialization of the network, the analyst can modify certain network files using the network definition language (NDL) so that all or a portion of the log-in sequence is omitted for TAF users. Log-in can be made completely automatic for any terminal. The analyst should refer to the NAM Network Definition Language Reference Manual for the procedures and files involved in providing this feature. In any case, the log-in sequence will be a subset of the following.

3. After a data path has been established, the network outputs the system banner and the request for family name. The two lines of the system banner appear with the following information.

yy/mm/dd. hh.mm.ss termnam
system header line

[†]The CR is the message terminator key. Depending on the terminal class, this may be the CR , SEND, or ETX key. In this manual, CR is used to indicate a message terminator.

^{††}Asynchronous line speeds which are recognized are 110, 150, and 300 baud for ASCII terminals and 110, 134.5, 150, 300, 600, and 1200 baud for the IBM 2741 terminal.

^{†††}Character sets which are recognized are ASCII, typewriter-paired ASCII APL, bit-paired ASCII APL, correspondence, Extended Binary Coded Decimal (EBCD), correspondence APL, and EBCD APL. The last four apply only to IBM 2741 terminals.

The first line consists of the date, the time, and a network-supplied terminal name, termnam. TAF references this terminal name when it sends information to the terminal, but it is of minimal importance to the user. The second line can contain the company name, the operating system name, or other information, depending on the individual site.

The request for family name appears on the next line, as follows:

FAMILY:

In response to this request, the user enters a preassigned family name (one to seven alphanumeric characters). If this family name is the system default family name, entering only Ⓔ is acceptable.

4. The network responds with a request for the user name. This appears as follows:

USER NAME:

The user enters a preassigned user name (one to seven alphanumeric characters).

5. The network responds with a request for password, as follows:

PASSWORD:
XXXXXXX

The terminal carriage is then positioned at the first character of the second line, when the hardware allows this. (The ability to position the carriage and overprint characters is not available on all terminals.) This allows a password to be entered without it being seen by anyone else. The user enters the password (four to seven alphanumeric characters) associated with the previously entered user name. (The released system requires that a password be associated with each user name and that it be at least four characters.)

6. The network might request the application name, as follows:

termnam - APPLICATION:

The termnam on this line is the same as that issued on the system banner. The user responds by entering:

TAF

(To the network, TAF is considered an application.) At this point the terminal is under the control of the transaction subsystem, which issues:

READY.

At this point the transaction subsystem is ready to receive input from the terminal.

Only one user can be logged in under a given user name at one time. When a user is logged in under a particular user

name, other terminals attempting to log in under that user name will receive the message CONNECTION REJECTED.

To avoid this lengthy dialog, it is possible to enter the family name, user name, password, and application name on one line, separated by commas in response to the request for the family name. All remaining prompts (except READY.) are suppressed, and input can then be sent to the transaction subsystem.

Because of the flexibility of the network and the transaction subsystem, the procedure to access the transaction subsystem can vary substantially from site to site. Therefore, the coordinator of transaction subsystem operations at each site should provide each user, or terminal location, with a detailed log-in procedure for that site.

The site should also provide information concerning terminal characteristics. A variety of terminal classes is supported for use on the network with TAF. Each of these terminal classes has characteristics, which the analyst can change by modifying the network files and which the user can change by entering terminal definition commands.

A discussion of the terminal characteristics, the defaults for each terminal class, and the procedure for changing these characteristics at a terminal are provided in appendix F.

IMPLEMENTING AN APPLICATION

The following steps specify the procedure for implementing an application.

1. Review this manual.
2. Design the application including:
 - Selection of the data manager (CDCS and/or CRM, TOTAL or TAF) to allow the correct choice of file types, file structure, and file interaction.
 - Selection of recovery and backup techniques, including journalizing of files.

Refer to the CYBER Database Control System Reference Manual, the TAF Data Manager Reference Manual, the CRM Data Manager Reference Manual, or the Total Reference Manual for more information (publication numbers are listed in the preface).

3. Describe and install the data base using the data manager. Refer to the appropriate data manager reference manual for this procedure.
4. Test load the data base using the data manager commands (usually done in batch mode).
5. Using the application design, write the application tasks to process transaction input.
6. Modify ITASK to reflect the actual application. Tasks MSABT, LOGT, and SYMSG may also require modification. (Refer to section 10.)

7. Build a task library. (Refer to section 9.)
8. Build an xxJ file. (Refer to section 4.)
9. Build a network file, and use NDL to configure terminals and establish log-in sequences and terminal characteristics.
10. Build the TAF configuration file (TCF). (Refer to section 4.)
11. Test the system.

SUBCONTROL POINTS

The following discussion is intended for analysts and may be omitted by the applications programmer.

Tasks reside at subcontrol points within the field length of the transaction executive, which resides at control point 2. Each task runs with its own exchange package, reference address (relative to the transaction executive), and field length (a subset of the field length of transaction executive). A task, therefore, can reside in any contiguous segment within the transaction executive field length. Memory is allocated in 100_g word blocks located on even 100_g word boundaries (for example, 32100_g to 32200_g). A subcontrol point consists of one 100_g word block of memory for system use, followed by sufficient blocks to contain an entire task.

The subcontrol point feature allows the transaction executive to control each task. Some of the advantages associated with the subcontrol points are:

- Isolating one subcontrol point from other subcontrol points and the transaction executive. This means that no application program can destroy the transaction executive or circumvent system security.

- Blocking RA+1 requests from a subcontrol point. No requests are allowed directly from a subcontrol point to NOS. Any such requests are intercepted by the system monitors, which return control to the transaction executive. Thus, the only RA+1 requests a task can legally issue are those processed by the transaction executive. This includes requests to the data manager.

The transaction subsystem allows a maximum of 31 subcontrol points. An installation parameter sets the number of control points that the transaction executive initializes. When the transaction executive is loaded, the operator can select a number of subcontrol points other than this default value. The number of subcontrol points must not be less than 2 nor greater than 31. Once the transaction subsystem is initialized, no change in the number of subcontrol points is allowed. The optimum number of subcontrol points is selected by the site.

Each subcontrol point requires eight words of table space. No space, other than a table entry, is allocated for a subcontrol point unless it is active. Each subcontrol point has 111_g words reserved, beginning at RA, similar to those used for batch control points.

MULTIMAINFRAME

Under a multimainframe configuration, the transaction subsystem can be run on either or both mainframes. However, a data base and JOUR0 file (section 4) cannot be shared by the transaction subsystems. Each subsystem must have a unique data base.

EXTENDED CORE STORAGE (ECS)

TAF does not allow the use of the user access to ECS feature. However, the TAF procedure file can declare data files and libraries as ECS-resident.

A task is isolated from the data manager and the remainder of the system by the transaction executive. The isolation guarantees that errors in a task do not propagate beyond the task. The following types of requests constitute the interface between the task and the transaction executive.

- Data manager requests
- Journal file requests
- Memory dump requests
- System requests
- Task scheduling requests
- Input/output requests[†]
- Task control requests
- Task utility requests

Internally, a task performs a request by loading address RA+1 of its subcontrol point. This activity is not visible to the application programmer who uses the macros or calls given in this section. The transaction executive uses the request parameters to determine what processing is required by the task. The address field and any fields in a parameter list that reference an address contained within the task are verified to ensure that the address referenced is within the task field length.

Tasks communicate with each other and accumulate input by using the communication block. The communication block is discussed next, followed by the types of requests.

COMMUNICATION BLOCK

Communication between a chain of tasks is provided by the communication block. The initial input or the input from a terminal as a result of the WAITINP (refer to Input/Output Requests later in this section) request is also stored in the communication block. The communication block can be used to pass parameters or data from task to task. The format of the communication block is shown in figure 2-1. The length of terminal input is not restricted by the length of the communication block. The LOADCB request is available to read extra input that overflows the communication block (refer to the description of i in figure 2-1). If the communication block is passed between tasks in a transaction and overflow data is present, other tasks in the transaction can access this data with a LOADCB request.

If COBOL is used, the communication block must be described in the COMMON-STORAGE SECTION. Shown here is an example of such a description:

DATA DIVISION.

COMMON-STORAGE SECTION.

01 COMMUNICATION-BLOCK.

03 CB-HEADER-AREA-WORD-ONE.

05 CB-DATA-BASE-NAME PIC X(2).

05 CB-USER-TST-AREA COMP-4 PIC 9(6).

05 CB-TRANS-SEQ-NUMBER COMP-4 PIC 9(6).

03 CB-HEADER-AREA-WORD-TWO.

05 CB-USERNAM PIC X(7).

05 CB-INPUT-STATUS-BYTE COMP-4 PIC 9.

05 CB-MESSAGE-WORD-COUNT COMP-4 PIC 99.

03 CB-MESSAGE-AREA.

03 CB-TRAILER-AREA.

05 FILLER PIC X(4).

05 CB-PACKED-DATE.

07 CB-YEAR COMP-4 PIC 9.

07 CB-MONTH COMP-4 PIC 9.

07 CB-DAY COMP-4 PIC 9.

05 CB-PACKED-TIME.

07 CB-HOUR COMP-4 PIC 9.

07 CB-MINUTE COMP-4 PIC 9.

07 CB-SECOND COMP-4 PIC 9.

WORKING-STORAGE SECTION.

•
•
•

This configuration will be referred to in this manual as the Standard COBOL Communication Block because of the correspondence between it and the fields shown and described in figure 2-1. However, applications programmers may find it convenient to use a more abbreviated communication block when a particular case allows.

COMP-4 is specified for data areas intended for numerical information. Refer to the COBOL 5 Reference Manual for information on the relationship between the number of digits specified in the picture clause for COMP-4 data and the number of bits reserved.

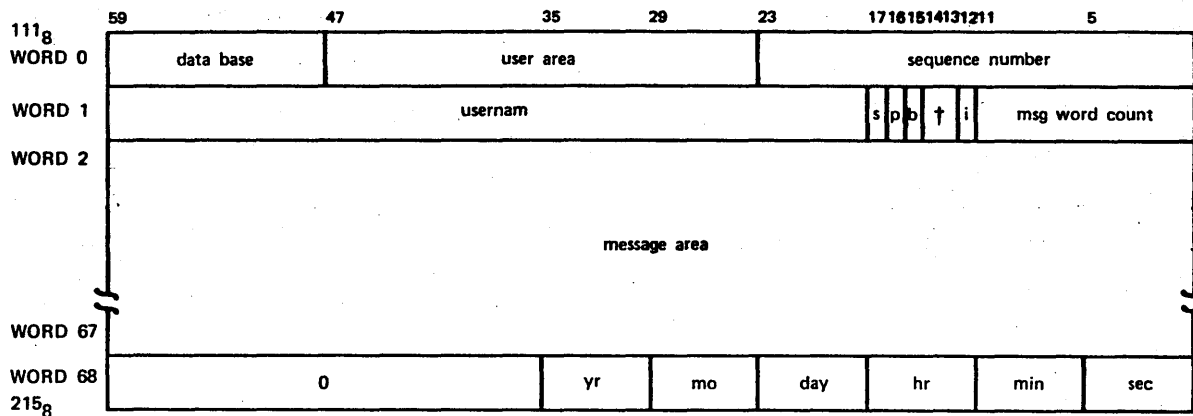
If FORTRAN is used, the communication block must be in the first block of labeled common.

If more data is input than can be contained in the communication block, the LOADCB request may be used to obtain the input.

REQUESTS

The following paragraphs describe the various types of requests a task can make. They describe task scheduling, input/output, task control, and task utility requests and show the formats for making these requests from tasks written in FORTRAN, COBOL, and COMPASS.

[†]The standard CRM-supported, compiler-generated I/O calls from FORTRAN and COBOL are not allowed. The COBOL SORT verb is not supported.



- data base** Data base the originating terminal is validated to use.
- user area** Image of originating terminal user area from the terminal status table (refer to section 7). Contains the recovery flag. (Refer to the TARO and TSIM requests for more information.)
- sequence number** Assigned by the transaction executive to each newly initiated transaction. This number is generated by incrementing a counter by 1 and identifies the transaction, or task chain, throughout its life in the subsystem.
- username** One- to seven-character user name, left-justified with binary zero fill from the terminal status table.
- s††** Bit 17 of word 1 is set to 1 if this communication block was created by a TAF-originated transaction. Word 2 (and possibly 3) of the communication block contains additional information (operating system or network defined conditions which may be processed by TAF-originated tasks). Refer to section 10 for information concerning required system tasks.
- p††** Bit 16 of word 1 is set to 1 if this communication contains data on which a parity error occurred during terminal input.
- b††** Bit 15 of word 1 is set to 1 if this communication block was created by a batch transaction (BTRAN request).
- i††** Bit 12 of word 1 is set to 1 if additional input to be loaded by means of the LOADCB request exists.
- msg word count** Length of the input message in words in the communication block proper.
- message area** A message from a terminal, as a result of initial terminal input, WAITNP, or BWAITNP requests, begins at word 2 and is terminated by 12 bits of zero in positions 11-0 of the last word of the message. All remaining message area words contain binary zero. The maximum length of a terminal message is 57 words. This area may also contain data from a BTRAN request. The maximum length of a BTRAN message is 62 words. In all cases, the length of the message is in the msg word count field. The information contained in this area is available to the user, and its contents is passed to any communication block generated by the processing of a CALLTSK or CALLRTN request. If it is desired to retain the terminal input, the user may only write in words msg word count+2 through 67. If there is no need to retain the terminal input, the entire area can be used.
- yr, mo, day, hr, min, sec** Date and time in packed binary format from the operating system PDATE macro.

Figure 2-1. Communication Block

† Reserved.

†† The main purpose of these fields is to notify the system task ITASK of certain conditions. Under COBOL, the EXTRACT routine allows examination of these fields. Also, a COBOL data type, computational-4, can be used to access these fields more easily; computational-4 data types are allowed only on COBOL version 5.2 and above. ITASK is described in section 11, and EXTRACT is described later in this section.

DATA MANAGER REQUESTS

A task can make data manager requests to one of the data managers, (CDCS, CRM TOTAL, or TAF) supported by the transaction subsystem. (Refer to the preface for the publications that document these data managers and the requests used.)

JOURNAL FILE REQUESTS

A task can supplement automatic journaling by writing data on a journal file. Journal files and the journal requests are discussed in section 4.

MEMORY DUMP REQUESTS

These requests are discussed in section 6.

SYSTEM REQUESTS

System requests from transaction tasks are limited to the following.

COMPASS Macros

ABORT

DATE x

ENDRUN

JDATE x

MEMORY (as described below)

MESSAGE x (to console B display, line 1 only)

PDATE x

RTIME x

TIME x

FORTRAN Functions and Subroutine Calls

DATE(x) or CALL DATE(x)

JDATE(x) or CALL JDATE(x)

TIME(x) or CALL TIME(x).

COBOL Statements

ACCEPT x FROM DATE.

ACCEPT x FROM TIME.

ACCEPT item FROM TTY.

DISPLAY item UPON TTY.

These requests are described in detail in volume 2 of the NOS Reference Manual, FORTRAN Extended 4 Reference Manual, and the COBOL 5 Reference Manual. (Refer to the preface for publication numbers.)

MEMORY REQUESTS (COMPASS ONLY)

Tasks are allowed to interrogate the status of and change their CM field length (FL) via the RA+1 requests MEM and RFL. The format of these calls is documented for the MEMORY macro in the NOS Reference Manual, volume 2. The use of these requests in TAF is restricted as follows:

- Memory requests should be issued with recall (r parameter); otherwise control may be returned before the request is processed.
- There is an upper limit beyond which a task is not allowed to increase its CM space; this limit is referred to as the maximum task field length (MFL). The MFL for each task is equal to the task FL plus the value specified by the EF LIBTASK directive for the task (refer to section 9). If a task attempts to exceed this value, it may be aborted (refer to documentation of the MEMORY macro NA parameter in the NOS Reference Manual, volume 2).
- Field length cannot be decreased to less than the last word address of the default communication block transfer address rounded up to the nearest 100g.
- A non-CM resident task making a memory request may be rolled out of CM if the request cannot be satisfied initially; therefore, a task must have no outstanding (incomplete) data manager requests.
- Since a task making a memory request may be rolled out, it is usually desirable to free locked data manager resources prior to the request.

Tasks will also be allowed to use the Common Memory Manager (CMM) with the following restrictions:

- The processing performed by CMM for the task must not result in CMM using the Fast Dynamic Loader or the task will be aborted.
- The above restrictions on use of MEM and RFL requests apply.

Common Memory Manager requests are documented in detail in the Common Memory Manager Version 1 Reference Manual.

TASK SCHEDULING REQUESTS

A task, including ITASK (refer to section 10), can call, in order, a list of other tasks. This process is referred to as task scheduling. The following four requests can be used to schedule tasks.

- CALLRTN
Calls up to five tasks. Upon completion of the called chain of tasks, returns control to the calling task.
- CALLTSK
Calls up to five tasks. The task making the call either ceases or continues to run, allowing the new task chain to proceed independently.

- **NEWTRAN**
Initiates up to five new tasks (COMPASS only).
- **TRANCHK**
Checks to see if a specific task is active (COMPASS only).

NOTE

If a list has either no tasks or more than five tasks, TAF aborts the task and issues the following error message on the screen.

TASK REQUEST ARGUMENT ERROR

addr Address of a list of from one to five task names. Each task name must be left-justified with binary zero fill, to a word boundary. The list is terminated by a word of binary zeros. Tasks are scheduled by CALLRTN in the order they appear on the list.

NOTE

If task_i, specified on a CALLRTN request, makes a CALLRTN request, task_i + 1 through task_n, specified on the original CALLRTN request, are not scheduled. (Refer to figure 2-4.)

CALLRTN Request

The CALLRTN request is used to request that the transaction executive schedule a task or series of tasks, with control returning to the calling task upon completion of the called tasks. A CALLRTN request cannot be made while data manager requests are outstanding. The current communication block is passed to the called tasks and returned to the caller when the caller is reinitiated. Upon return, the caller begins execution immediately after the CALLRTN request. Only the information in the communication block may have changed. Return from the called tasks to the caller takes place automatically after the last called task ceases. A task waiting for return may be temporarily rolled from memory.

The format is:

COBOL

ENTER CALLRTN USING task₁ task₂...task_n.

task_i Parameter whose value is a one- to seven-character alphanumeric task name. The tasks are scheduled by CALLRTN in the order specified. From one to five tasks can be specified.

FORTRAN

CALL CALLRTN(task₁,task₂,...,task_n)

task_i Parameter whose value is a one- to seven-character alphanumeric task name, left-justified with either blank or binary zero fill to a word boundary. The tasks are scheduled by CALLRTN in the order specified. From one to five tasks can be specified.

COMPASS

CALLRTN addr

The following diagrams illustrate CALLRTN processing.

In figure 2-2, TASK A, at subcontrol point X, executes a CALLRTN to TASK B. TAF will first load a copy of TASK B to an available subcontrol point (unless a serially reusable copy of TASK B is already at a subcontrol point or CM resident). TAF will then copy the contents of TASK A's communications block (C.B.) to TASK B's communication block area. TASK B begins execution. When TASK B has completed execution, TAF will copy the communications block (which was probably modified by TASK B) back to TASK A and allow TASK A to continue execution at the instruction immediately following the CALLRTN command.

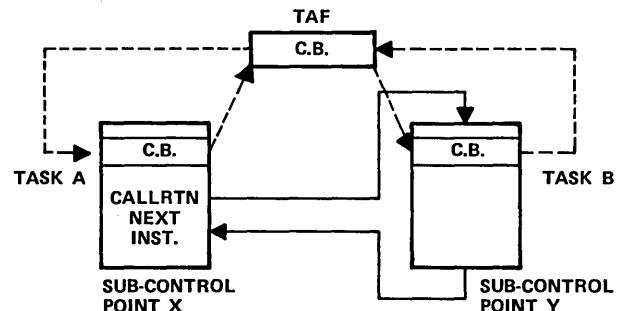


Figure 2-2. CALLRTN

The CALLRTN command allows a list (chain) of up to five tasks to be called. Figure 2-3 shows TASK A calling a chain of three tasks. As each task in the chain is completed, TAF will automatically pass the communications block to the next task in the chain and allow that task to execute. When the last task in the chain has completed, TAF will pass the communications block back to the calling task and allow it to continue execution at the instruction immediately following the CALLRTN command.

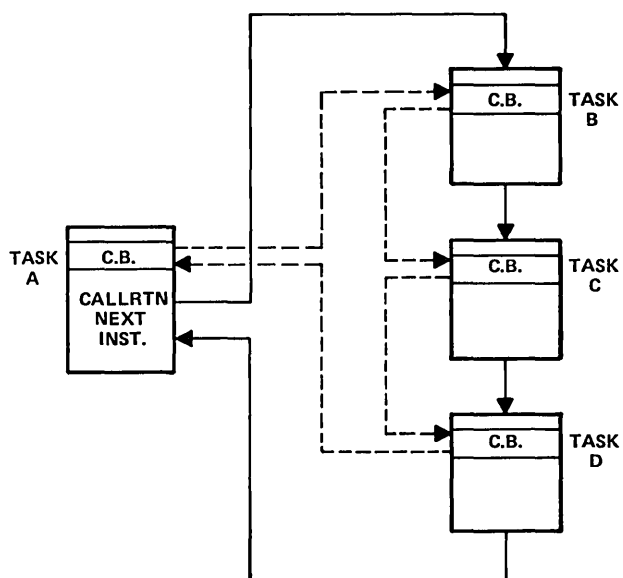


Figure 2-3. CALLRTN (CHAIN)

Figure 2-4 illustrates the processing referred to in the note above. The majority of the data inputs are handled by the TASK B-C-D chain. When an exception is found in TASK C, exception processing is invoked in the TASK E-F chain. Each new CALLRTN command cancels the previous one (TASK D is not executed), but TAF will always bring control back to the instruction immediately following the original CALLRTN command (TASK A).

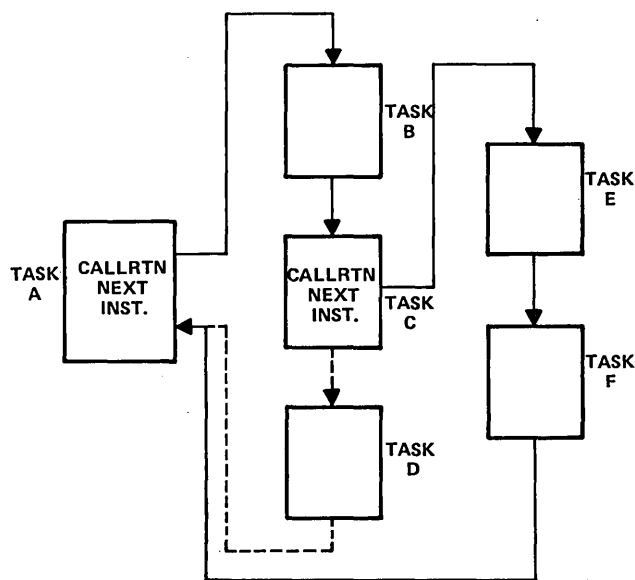


Figure 2-4. CALLRTN (MULTIPLE CHAINS)

CALLTSK Request

The CALLTSK request is used to request that the transaction executive schedule a task or series of tasks (task chain). Included in this request is a parameter option which enables task branching or a CEASE function to be performed after the call is honored. If the task is not to cease on this call, a branching of the chain initiates a new chain which is independent of the original chain. The new chain is assigned a new sequence number for unique identification within the transaction subsystem (figure 2-1). The number of branches allowed by a transaction is controlled by an assembly constant MAXBW.

The format is:

COBOL

ENTER CALLTSK USING task code task₁...task_n.

task Parameter whose value is a one- to seven-character alphanumeric task name. This task is the next task to call in the present task chain (same sequence number) if code is set to zero; otherwise, it is the first task of a branch chain (the called task gets a different sequence number and proceeds independently of the calling task).

code Computational-1 item whose value is nonzero if the calling task is to continue after the request, initiating an independent task chain. If the value of this parameter is 0, the calling task ceases after this request. This parameter must be specified.

task_i Parameter whose value is a one- to seven-character alphanumeric task name. From zero to four tasks can be specified. These tasks are scheduled after the one specified in the task parameter.

FORTTRAN

CALL CALLTSK(task,code,task₁,...,task_n)

task Parameter whose value is a one- to seven-character alphanumeric task name, left-justified with either blank or binary zero fill to a word boundary. This task is the next task to call in the present task chain (same sequence number) if code is set to zero; otherwise, it is the first task of a branch chain (the called task gets a different sequence number and proceeds independently of the calling task).

code Integer whose value is nonzero if the calling task is to continue after the request, initiating an independent task chain. If the value of this parameter is 0, the calling task ceases after this request. If not specified, the default is 0.

task; Parameter whose value is a one- to seven-character alphanumeric task name, left-justified with either blank or binary zero fill to a word boundary. From zero to four tasks can be specified. These tasks are scheduled after the one specified in the task parameter.

COMPASS

CALLTSK addr,ceas

addr Address of a list of from one to five task names. Each task name must be left-justified with binary zero fill to a word boundary. The list is terminated by a word of binary 0's. CALLTSK schedules the tasks in the order they appear on the list.

ceas If specified, the calling task ceases after this request. If not specified, the calling task continues after the request. In this case the CALLTSK initiates an independent task chain.

The following diagrams illustrate CALLTSK processing.

Figure 2-5 shows TASK A making a CALLTSK with CEASE to TASK B. TAF will load a copy of TASK B to an available subcontrol point (this step is not necessary if a serially reusable copy of TASK B is already at a subcontrol point or CM resident). The communications block will be copied from TASK A to TASK B. TASK A's subcontrol point is released for use by other transactions, and TASK B is allowed to begin execution.

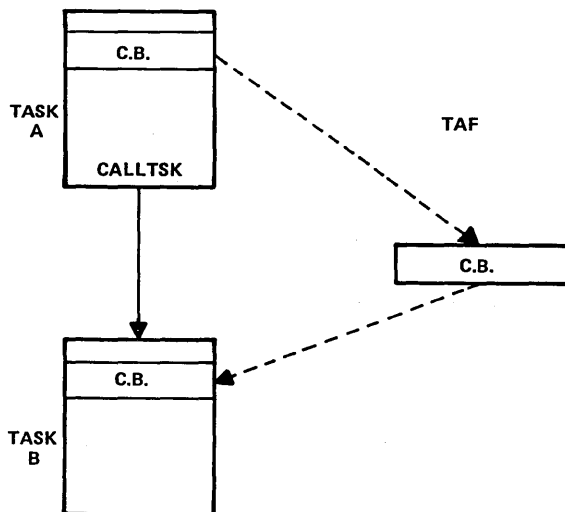


Figure 2-5. CALLTSK With CEASE

The CALLTSK without CEASE command allows a new asynchronous chain to be started. In figure 2-6, TASK A uses this command to call TASK B. TAF will copy the contents of TASK A's communications block to a new communication block. A copy of TASK B will be loaded to an available subcontrol point (unless a serially reusable copy of TASK B is already at a subcontrol point or CM resident). The new communications block will be copied to TASK B. TASK A is allowed to continue execution and TASK B is allowed to begin execution.

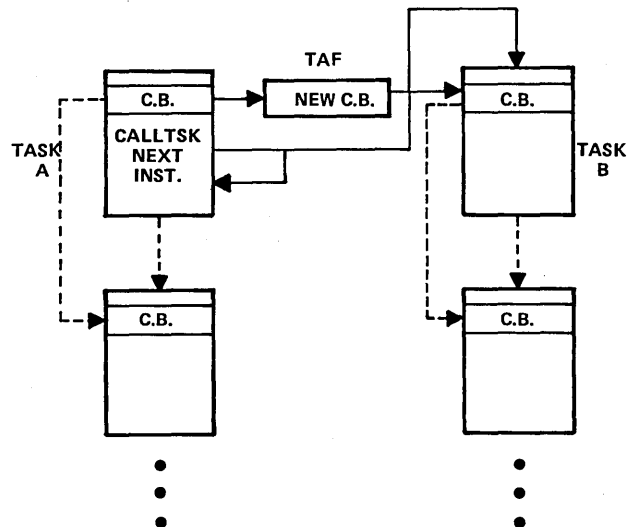


Figure 2-6. CALLTSK Without CEASE

NEWTRAN Request

The NEWTRAN request initiates a new transaction chain with a unique sequence number and with the data base, user area, and user name set to 0 in the communication block. The NEWTRAN request is designed for system use, primarily by ITASK, and is provided only for COMPASS.

The format is:

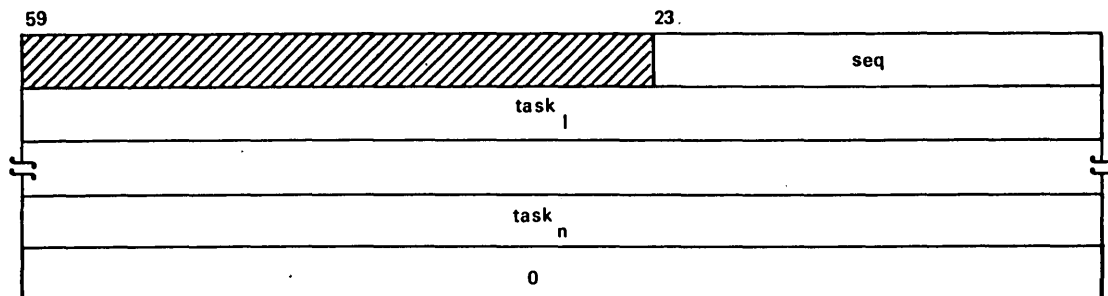
COMPASS

NEWTRAN addr

addr Address of a parameter table in the form shown in figure 2-7.

TRANCHK Request

The TRANCHK request determines the status of a transaction. This request is provided only for COMPASS.



seq The transaction executive returns a new 24-bit binary transaction sequence number in this field.

task_i One- to seven-character task name, left-justified with binary zero fill. From one to five tasks can be specified; the list is terminated by a word of binary zeros. NEWTRAN schedules the tasks in the order they appear on the list.

Figure 2-7. NEWTRAN Parameter Table

The format is:

COMPASS

TRANCHK addr

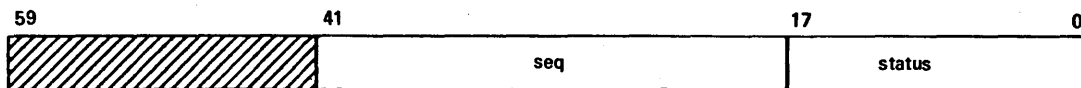
addr Address of a parameter table in the form shown in figure 2-8.

INPUT/OUTPUT REQUESTS

The following requests are associated with input/output.

- **WAITINP**
Places terminal input data in the communication block. Equivalent to a read request.
- **LOADCB**
Reads data that overflowed the communication block.
- **SETCHT**
Sets character type for terminal input.
- **BLDABH**
Builds the application block header to use different character types, terminal format effectors, and auto input.

- **GETABH**
Returns the last application block header to the task.
- **TERMDEF**
Defines the operational controls or format of the terminal.
- **SEND**
Sends data to terminal. Equivalent to a write request.
- **RELSCB**
Releases memory used for extra communication blocks.
- **TSIM**
Reads terminal status table.
- **TARO**
Writes into terminal status table.



seq Binary transactions sequence number which identifies the transaction (task or task chain).

status The system returns one of the following as status in this field.

- 1 The task chain is still active.
- 0 The task chain is inactive.

Figure 2-8. TRANCHK Parameter Table

- **IIO**

Limits total number of requests (RA+1 calls) a task can make.

- **BWAITINP**

Specifies that terminal input data is to be placed in a location in the task field length that is not reserved for the communication block.

- **TPSTAT**

Returns a value which indicates whether the active terminal subsystem is TS or NAM.

- **BEGIN**

Specifies the area where the terminal transmission is to be placed in the task field length. It does not return the communication block to the first word address of the task unless the task requests.

from the original task chain must follow a message received following a WAITINP request. A task that includes a WAITINP request must not have been called by a CALLRTN request.

No data manager requests may be active when a task makes a WAITINP request. Thus, for transactions using the TAF data manager, a RECALAL (refer to the RECALAL request later in this section) should precede a WAITINP if there is any chance that data manager requests may be outstanding. All tasks that make WAITINP requests must be entered in the task library with the scheduling queue limit set to 1. (Refer to section 9.)

WAITINP does not cause any data manager cease requests to be issued. Therefore, any resources locked or reserved by a transaction prior to a WAITINP, continue to be locked or reserved during and after the WAITINP.

WAITINP Request

The WAITINP request (wait for terminal input) reads data from a terminal. Internally, this request informs the transaction executive that the next input received from a terminal is to be passed directly to the requesting task and is not the initiation of a new transaction. To a task, the data appears in the communication block. The data held in the communication block prior to the WAITINP is lost.

The first parameter is a data item or variable whose value the system sets to indicate the reception of terminal input or an error condition. An optional second parameter of the request specifies a time limit, after which processing of the transaction is restarted even if no input has been received from the terminal.

NOTE

At least one message (SEND) must be sent to the terminal before making a WAITINP request, and at least one SEND request

When a WAITINP request is processed, all communication blocks are released and the task is rolled out. All data in the communication block, except the transaction sequence number, will be lost, so the user should save any data to be preserved prior to the request. When input arrives from the terminal, a new communication block is generated, and the task is restarted. The new communication block will contain the transaction sequence number associated with the task prior to the WAITINP request. If the request does not return a normal status, the information in the communication block is meaningless.

The format is:

COBOL

ENTER WAITINP USING status time.

status Computational-1 item which the system sets to indicate the status of the WAITNP request. Upon completion of the request, stat has one of the following values.

- 0 The request processed normally; terminal input appears in the normal input portion (begins in the third word) of the communication block.
- < 0 Another task is waiting for input from the originating terminal. The WAITNP request must be reissued if input is desired.
- > 0 The time delay elapsed before input was received from the terminal.

time Computational-1 item which contains the time limit, in decimal seconds, to wait for input. If not specified, the default is 480 seconds. The maximum value is 2048 seconds.

FORTTRAN

CALL WAITNP(status,time)

status Integer variable which the system sets to indicate the status of the WAITNP request. Upon completion of the request, stat has one of the following values.

- 0 The request processed normally; terminal input appears in the normal input portion (begins in the third word) of the communication block.
- < 0 Another task is waiting for input from the originating terminal. The WAITNP request must be reissued if input is desired.
- > 0 The time delay elapsed before input was received from the terminal.

time Integer time limit, in decimal seconds to wait for input. If not specified, the default is 480 seconds. The maximum value is 2048 seconds.

COMPASS

WAITNP time

time Time limit, in decimal seconds to wait for input. If not specified, the default is 480 seconds. The maximum value is 2048 seconds.

Status of the WAITNP macro is determined by the following:

- Word 2 of the communication block contains 1 if the time delay elapsed before input was received from the terminal.
- The X6 register contains a negative value if another task is waiting for input from the originating terminal. The WAITNP request must be reissued if input is desired. (Word 2 of the communication block should be checked before X6 is checked.)
- The X6 register is set to 0 if the request processed normally; terminal input appears in the normal input portion (begins in the third word) of the communication block. (Word 2 of the communication block should be checked before X6 is checked.)

LOADCB Request

If the transaction input data exceeds the message area of the communication block, bit 12 of word 1 of the communication block (the i field) is set. The LOADCB request loads the remainder of the input message into a user-defined buffer. This buffer is a single contiguous block of storage for all additional communication blocks. No headers or trailers appear in this area.

The maximum input size is determined by an installation parameter.

The data that remains after the communication block message area fills is transferred to the user-defined buffer. If the LOADCB request is made with no data remaining (i field not set) or if a RELSCB request was made, no data is transferred and the status word is set to 0. If the remaining data consists of more words than will fit in the buffer, the buffer is filled, and the status word is set to a value greater than the number of words in the buffer. If the remaining data can be contained in the buffer, the remaining data is transferred to the buffer, and the number of words transferred is set in the status word.

The message data exceeding the communication block can be requested as many times during a transaction as desired. Each time, the overflow data is transmitted as specified previously.

The format is:

COBOL

ENTER LOADCB USING buff status r.

buff A 01-level item identifying the user-defined buffer area.

status Computational-1 item which the system sets to indicate the LOADCB request status. The significance of the value set has been described earlier in the LOADCB request description.

r Computational-1 item. If a nonzero value is entered for this parameter, any message data that overflows the LOADCB buffer specified with the buff parameter is released (same as RELSCB).

FORTRAN

CALL LOADCB(buff,len,status,r)

buff An array identifying the user-defined buffer area.

len Integer length of the array specified with the buff parameter in words.

status Variable which the system sets to indicate the LOADCB request status. The significance of the value set has been described earlier in the LOADCB request description. This value is also returned as the value of the function, if LOADCB is called as a FORTRAN function.

r Integer variable. If a nonzero value is entered for this parameter, any message data that overflows the LOADCB buffer specified with the buff parameter is released (same as RELSCB).

As stated in the description of the stat parameter, the LOADCB request can be referenced as a FORTRAN function. The format of the function is LOADCB(buff,len,stat,r); the parameters are the same as those already described. The value of the function is the value of the stat parameter.

COMPASS

LOADCB addr

addr Address of a parameter table in the format shown in figure 2-9.

The LOADCB macro status is returned in the X6 register. The significance of the value returned has been described earlier in the LOADCB request description.

SETCHT Request

NOTE

This is an optional request; the default character type is 6-bit display code characters.

The SETCHT request allows a task to change the application character type (ACT) of terminal messages placed in the communication block. The following character types are available.

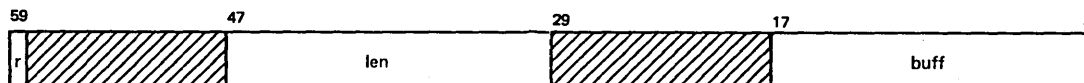
- 8-bit ASCII, seven and one-half characters per word.
- 8-bit ASCII characters, right-justified in 12-bit bytes, five characters per word.
- 6-bit display code characters, 10-characters per word.

This request causes TAF to send a change input character type supervisory message to NAM. For more information about supervisory messages, refer to the Network Access Method Reference Manual.

NOTE

This request does not change the data currently held in communication blocks for the task. For example, a LOADCB request gets data in the former character type, while the next WAITNP request gets data in the new character type.

If ASCII characters are used to originate transactions, ITASK transaction tables must be set up in terms of ASCII and not display.



r If bit 59 is set to 1, any message data that overflows the LOADCB buffer is released (same as RELSCB).

len Length of the LOADCB buffer in words.

buff First word address of the buffer area.

Figure 2-9. LOADCB Parameter Table

The format is:

COBOL

ENTER SETCHT USING usernam stat act.

usern A 01-level item containing a one- to seven-character user name, left-justified with blank fill. The user name may also be specified with a computational-1 item with binary zero fill. If the item contains only binary zeros, the character type applies to the originating terminal. This user name must be a name appearing in the NCTFid file. A user name consisting of the character P with binary zero fill cannot be specified.

stat Computational-1 item whose value is set by TAF to indicate user name status. This value is zero if the user is not logged in and one if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the specified user does not match the data base of the originating user.

act A 01-level item containing one of the following words, left-justified.

 ASCII7 8-bit ASCII. Seven and one-half characters per word.

 ASCII5 8-bit ASCII, right-justified in 12-bit bytes, five characters per word.

 DISPLAY6-bit display code.

FORTRAN

CALL SETCHT(usernam,stat,act)

usern One- to seven-character user name, left-justified with zero or blank fill. If the terminal name is 0, the character type applies to the originating terminal.

stat Unnormalized floating-point number whose value specifies the user name status. This value is zero if the user is not logged in and one if the request is processed normally.

 The task is aborted if the user name is not defined or if the data base for the specified user does not match the data base of the originating user.

act Parameter containing one of the following words, left-justified with binary zero or blank fill.

 ASCII7 8-bit ASCII, seven and one-half characters per word.

 ASCII5 8-bit ASCII, right-justified in 12-bit bytes, five characters per word.

 DISPLAY6-bit display code.

COMPASS

SETCHT addr

addr Address of a parameter table in the form shown in figure 2-10.



usern One- to seven- character user name, left-justified with zero fill. If the user name is 0, the character type applies to the originating terminal.

ty Binary value for character type:

 2 8-bit ASCII, seven and one-half characters per word

 3 8-bit ASCII, right-justified in 12-bit bytes.

 4 6-bit display code.

stat Address of a word in which TAF returns an unnormalized floating-point number to indicate the user status. This value is zero if the user is not logged in and one if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the specified user does not match the data base of the originating user.

Figure 2-10. SETCHT Parameter Table

BLDABH Request

NOTE

This is an optional request. The default application block header (ABH) specifies 6-bit display code characters, nontransparent mode, no format effectors, and no auto-input.

The possible values are listed in table 2-1.

value_i

A computational-1 item whose value is set for the corresponding field_i. The field_i and value_i parameters must appear in pairs. The possible values and descriptions are listed in table 2-1.

The BLDABH request builds or modifies the ABH. This header accompanies every block passing between TAF and NAM, and thus is used during communication between TAF and terminals. It contains detailed information describing the block it accompanies. To use some of the NAM features, a task must use the BLDABH request to modify the ABH. Some of the features provided via the ABH are:

- Character type specification of messages sent to a terminal. Either ASCII or display code characters may be used.
- Transparent mode specification. Transparent mode inhibits all data transforms and allows the task direct access to the real terminal features.
- Terminal format effector specification. Format effectors control printing (or the cursor on display terminals).
- Auto-input mode specification. Auto-input mode allows a fill in the form type of transaction. This is provided since the first n characters of a message output to the terminal via the SEND request (ABT=2 only) can be used to form the first n characters of subsequent input from the terminal. If n is greater than 20, only the first 20 characters are returned, followed by the entered text.

It is not necessary to use the BLDABH request before every SEND. It is needed when a selectable parameter on the ABH for the next SEND differs from a setting on the ABH for the previous SEND. In other words, a setting of a parameter once selected by BLDABH stays in effect for that task unless changed by a subsequent BLDABH request. If a task issues a BLDABH request and then calls another task, the new task must also issue a BLDABH request because the request parameters are not passed to it. If a task is to be reusable, the task should initialize the ABH if the default values have been changed.

Default values are supplied on the ABH. If the default values are the desired values, no BLDABH request is required. Default values have been selected to remain compatible with prior versions of the transaction subsystem.

The format is:

COBOL

ENTER BLDABH USING field₁ value₁...field_n value_n.

field_i A 01-level item containing a three-character left-justified field, for which a value is to be set. field_i can be binary zero or blank filled.

TABLE 2-1. BLDABH REQUEST PARAMETERS

Field _i	Value _i	Description
ACT		Application character type.
	2	8-bit ASCII characters, seven and one-half per word.
	3	8-bit ASCII characters, right-justified in 12-bit bytes, five per word.
NFE	4 [†]	6-bit display code characters, 10 per word.
		Format effectors in message.
	0	Format effectors appear in the message. (Refer to the NAM Reference Manual.)
AIM	1 [†]	No format effectors in message.
		Auto-input mode.
	0 [†]	No auto-input mode.
XPT	1	Auto-input mode selected.
		Transparent mode. Applies only to ASCII character types.
	0	Interactive virtual terminal (IVT) data transforms to console devices are performed. (This option must be selected for output to batch devices.)
	1	IVT transforms are inhibited.

[†]These are default values supplied by TAF, FORTRAN, and COBOL library routines for BLDABH and SEND. These are also the default values for the COMPASS SEND request when the task does not specify the ABH.

FORTRAN

CALL BLDABH(field₁,value₁,...,field_n,value_n)

field_i Variable containing a three-character left-justified field, for which a value is to be set. field_i can be binary zero or blank filled.

The possible values are listed in table 2-1.

value_i Integer which is set for the corresponding field_i. The field_i and value_i parameters must appear in pairs. The possible values and descriptions are listed in table 2-1.

Example:

The following request builds the ABH with ACT=4, NFE=0, and AIM=1.

```
CALL BLDABH(3LACT,4,3LNFE,0,3LAIM,1)
```

COMPASS

The COMPASS user specifies changes to the ABH with the SEND request (described later in this section).

GETABH Request

NOTE

This is an optional request. Terminal input/output can be done without using it.

The GETABH request allows a task to examine the ABH that accompanied the last message. If no SEND has been issued since the last input from the terminal, the GETABH request allows the task to obtain the ABH that accompanied the input block from the network. If a SEND has been issued, the ABH used for the SEND is obtained. This request is useful for obtaining the character type, character count, and parity error fields.

NOTE

The parity error indicators can also be obtained from the communication block, but it may be easier for the COBOL task to use GETABH to obtain this indicator.

The format is:

COBOL

```
ENTER GETABH USING usernam stat field1
value1...fieldn valuen.
```

usern_{am} A 01-level item containing a one- to seven-character user name, left-justified with blank fill. The user name may also be specified with a computational-1 item with binary zero fill. If the item contains only binary zero, the ABH for the originating terminal is returned. This

user name must be a name appearing in the NCTFid file. A user name consisting of the character P with binary zero fill cannot be specified.

stat Computational-1 item whose value is set by TAF to indicate the status of the user name. stat contents is 0 if the user is not logged in and is 1 if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the user name does not match the data base of the originating user.

field_i A 01-level item containing a three-character field for which a value is desired, left-justified with binary zero or blank fill. The possible fields and values are described in table 2-2.

value_i Computational-1 item in which the value for the desired field is to be returned. The field_i and value_i items must appear in pairs.

FORTRAN

```
CALL GETABH(usernam,stat,field1,value1,...,fieldn,
valuen)
```

usern_{am} Parameter whose value is a one- to seven-character user name, left-justified, binary zero, or blank filled. If the user name is binary zero, the ABH for the originating user is returned.

stat Unnormalized floating-point number whose value is set by TAF to indicate the user name status. stat contents is 0 if the user is not logged in and greater than 0 if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the user name does not match the data base of the originating user.

field_i Variable containing left-justified three-character field for which a value is desired. The possible fields and values are described in table 2-2.

value_i Name of an unnormalized floating-point variable in which TAF returns the value of the desired field.

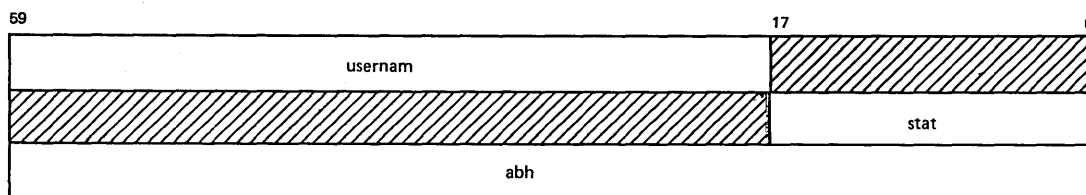
COMPASS

```
GETABH addr
```

addr Address of a parameter table in the format shown in figure 2-11.

TABLE 2-2. APPLICATION BLOCK HEADER

Field _i	Bits	Value _i	Description	Field _i	Bits	Value _i	Description
ABT	59-54	0	Application block type. Null input block.	XPT	14	0	Transparent bit. Nontransparent data; IVT transforms are used.
		1	Data block; not the last (applies to SEND only).			1	If application character type (ACT) is set to 2 or 3, IVT transforms are inhibited. If ACT is set to 4, this bit is ignored on output.
		2	Data block; last or only block.				
ADR	53-42		Addressing information; TAF assigns this area for output, NAM for input.				
ABN	41-24		Application block number. This area is user or TAF assigned for SEND requests. The user assigns this area causing the block parameter on the SEND request.				
ACT	23-20		Application character type.	BIT	12	0	For input, if XPT is set to 1 and ACT is set to 4, TAF changes ACT to 3. The user must change ACT back to 4 when no more transparent input is expected.
		2	8-bit ASCII characters, seven and one-half per word.			1	For input, parity error flag; for output, auto-input mode flag.
		3	8-bit ASCII characters, right-justified in 12-bit bytes, five per word.				
		4	6-bit display code characters, 10 per word.	TLC	11-0		
NFE	15		Format effectors.				
		0	Format effectors are present in the SEND message.				
		1	Format effectors are not present in the SEND message.				



usernam One- to seven-character user name, left-justified with binary zero fill. If the user name is 0, the ABH for the originating terminal is returned.

stat Address of a word in which TAF returns an unnormalized floating-point number to indicate the user name status. stat is 0 if the user is not logged in and 1 if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the user name does not match the data base of the originating user.

abh Application block header. The format and description of this word is listed in table 2-2.

Figure 2-11. GETABH Parameter Table

TERMDEF Request

NOTE

This is an optional request. Terminal input/output can be done without it. Each terminal that logs into the network is associated with a default set of terminal characteristics. (Refer to appendix F.)

The TERMDEF request can be used by a task to define or modify terminal characteristics, where those of the connected real terminal differ from the assumptions made for the network's IVT model. This request can also be used to tailor the operational controls of the terminal. A complete list of terminal characteristics is provided in the NAM Reference Manual and appendix F. Characteristics that may be modified include:

- Terminal physical page width.
- Number of physical lines on a terminal page.
- Specification of a character used to cancel an input logical line in progress.
- Specification of a character to be used as a break character. User breaks are sent to the application task for processing.
- Set transparent input mode. This setting exists for only one logical input line.

The format is:

COBOL

ENTER TERMDEF USING usernam stat field₁
field₂...field_n.

usern	A 01-level item containing a one- to seven-character user name left-justified with blank fill. The user name may also be specified with a computational-1 item with binary zero fill. If the item contains only binary zero, the terminal definition specified applies to the originating terminal. This user name must be a name appearing in the NCTFid file. A user name consisting of the character P with binary zero fill cannot be specified.
stat	Computational-1 item which TAF sets to indicate user name status. stat contents is 0 if the user is not logged in and is 1 if the request is processed normally. The task is aborted if the user name is not defined or if the user's data base does not match the data base of the originating user.

field_i

A left-justified 01-level item with binary zero or blank fill, containing the characteristic (for example, PW=60, PL=35, or CN=2B). A maximum of 17 nonblank, nonbinary zero characters is allowed. The characteristics are described in the NAM Reference Manual and appendix F of this manual. If the NAM Reference Manual specifies x or h, a display code alphanumeric representation is required (where x is an ASCII character and h is a hexadecimal character).

FORTRAN

CALL TERMDEF(usernam,stat,field₁,field₂,...,field_n)

usern	Parameter whose value is a one- to seven-character user name, left-justified with binary zero or blank fill. If the user name is 0, the terminal definition specified applies to the originating terminal.
-------	--

stat	Variable in which TAF places an unnormalized floating-point number to indicate user name status. stat contents is 0 if the user is not logged in and greater than 0 if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the user does not match the data base of the originating user.
------	--

field_i

A left-justified variable with binary zero or blank fill, containing the characteristic (for example, PW=60, PL=35, or CN=2B). A maximum of 17 nonblank, nonbinary zero characters is allowed. The characteristics are described in the NAM Reference Manual and appendix F of this manual. If the NAM Reference Manual specifies x or h, a display code alphanumeric representation is required (where x is an ASCII character and h is a hexadecimal character).

Example:

The following request designates the page width to be 60 characters, the page length to be 40 lines, and the plus character (ASCII representation is 2B) to be the cancel character for user USER5; it then assigns the status to variable STATUS.

```
CALL TERMDEF(5HUSER5,STATUS,5LPW=60,  
5LPL=40,5LCN=2B)
```

Example:

The following statements redefine the transparent input mode text delimiter to be a slant (ASCII representation is 2F) and the character count delimiter to be 4096.

```
DATA MSG/10HDL=X2F,C40,2L96/
.
.
.
CALL TERMDEF(5HUSER5,STATUS,MSG)
.
.
.
END
```

COMPASS

TERMDEF addr

addr Address of a parameter table in the format shown in figure 2-12.

SEND Request

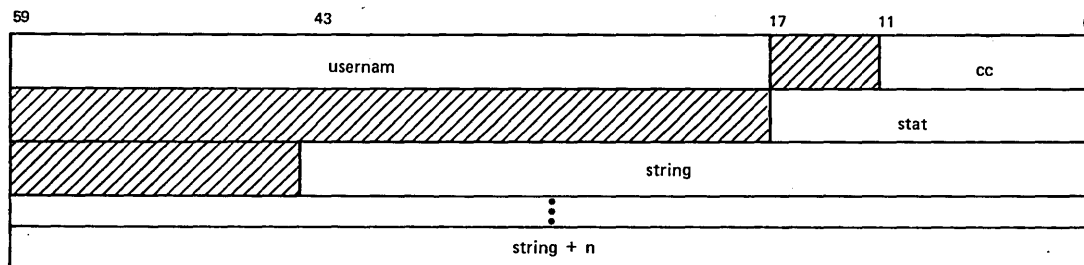
The SEND request sends a message to a user; that is, it performs a write operation. If the task does not specify the user to which the message should be sent, it is directed to the terminal that originated the transaction. At least one SEND request to the originating terminal must occur from the primary task chain for each message received. The task can also specify that the task be ended when the message is sent.

There is both a limit on the total output that one task can send (MAXTO), and a limit on the output one task can send before it is rolled out, pending the transfer of the output to the terminal or terminals (MLIN).[†] Also, the maximum output length for one SEND request is determined by the MAXWS[†] installation parameter.

A task can specify the stat parameter in COBOL and FORTRAN or the r parameter in COMPASS so that the task will wait until the network acknowledges that the message has been delivered to the user. If recall is selected, any error detected on the output message causes the task performing the SEND request to be restarted. Therefore, tasks performing SEND requests with recall should check stat and compensate for errors. Use of recall can simplify recovery since the task issuing the SEND request is informed of the problem at the first address after the SEND. When recall is selected, the task is rolled out if the message length exceeds the MLIN installation parameter. Any data manager resources held by the task, such as locked records and buffer space for currently held records, remain assigned for the duration of the rollout. If reserving these resources for this period of time will impact the performance of the application, then tasks using the TAF data manager should issue a CLEAR^{††} request to return all data manager resources before executing the SEND request.

If recall is not selected, ITASK is informed of the problem, rather than the task that issued the request. ITASK is informed because the issuing task may have already terminated when the message delivering problem is detected.

The format of the message on the terminal depends on the terminal class and the format effectors in the message. For example, on synchronous terminals the cursor for entering the next message to the host (upline) is always the first character of the line. On asynchronous terminals the



usernাম One- to seven-character user name, left-justified with binary zero fill. If the user name is 0, the terminal definition specified applies to the originating user.

cc The number of 8-bit ASCII characters (at seven and one-half characters per word) available in string + 2. Adding 2 is necessary because string begins in bit 43. If no characters are specified in the string, cc should equal 2.

stat An address of a word in which TAF returns the user name status in unnormalized floating-point format. The stat contents is 0 if the user is not logged in and greater than 0 if the request is processed normally. The task is aborted if the user name is not defined or if the data base for the user does not match the data of the originating user.

string A string of 8-bit ASCII characters, in hexadecimal notation, left-justified in the field, defining the terminal characteristic. (Refer to the NAM Reference Manual for a list of these characteristics.) As an example, to change the cancel key to a +, the string CN=+ is written as 434E3D2B.

Figure 2-12. TERMDEF Parameter Table

[†]Refer to the NOS Installation Handbook (publication number is contained in preface).

^{††}Refer to the TAF Data Manager Reference Manual (publication number is contained in preface).

cursor follows the last character of output; this assumes no format effectors are in the message. The NAM Reference Manual contains complete information on terminal protocols.

The format is:

COBOL

ENTER "SEND" USING mesaddr usernam cease-flag
output-flag block stat length.

The parameters are described in table 2-3.

NOTE

Under COBOL, SEND must be enclosed in quotation marks, because SEND is a reserved word for that compiler.

FORTTRAN

CALL SEND(mesaddr,length,usernam,cease-flag,
output-flag,block,stat)

The parameters are described in table 2-4.

COMPASS

SEND addr

addr

Address of a parameter table in the format shown in figure 2-13. This figure and the parameters in table 2-5 should be used with table 2-6. This table shows interrelationships between SEND parameters. Setting parameters shown in the top portion of table 2-6 results in the corresponding events shown in the bottom portion.

TABLE 2-3. COBOL SEND REQUEST PARAMETERS

Parameter	Data Type	Value	Description
mesaddr	01-level display item		Data item containing the message to be sent. For ASCII character types, the message must have an end-of-line byte of the single character US for all logical lines. (Refer to the NAM Reference Manual for NAM data formats.) The TAF SEND interface supplies end-of-line bytes for display code character types only.
usernam	01-level display item, or computational-1		Data item containing a one- to seven-character user name. If this parameter is omitted or specified as Comp-1 Value zero, the message is sent to the originating terminal. This user name must be a name appearing in the NCTFid file and must be validated for the same data base as the sending task. [†] A user name consisting of the character P with binary zero fill cannot be specified. The user name can be binary zero or blank filled.
cease-flag	Computational-1	0, omitted 1	Task end flag. The task continues after the message is sent. The task ends after the message is sent.

TABLE 2-3. COBOL SEND REQUEST PARAMETERS (Contd)

Parameter	Data Type	Value	Description
output-flag	Computational-1	0, omitted	Application block type. Indicates that this is the last or only data block in the message.
		1	Indicates that this is not the last data block in the message. (If both cease-flag and output-flag are set to one, there must be another task in the task chain.)
block	Computational-1	Nonzero	Application block number data name. TAF uses the value contained in the data name as the ABN.
		0	TAF assigns the ABN and returns this as the value in this data name.
		Omitted	TAF assigns the ABN, and the value is not returned.
stat	Computational-1		Recall flag; this parameter applies only if the cease-flag field is set to 0.
		0, omitted	The task begins processing after the block is transmitted to the network.
		Nonzero	The task begins processing after the block has been delivered to the user and acknowledgment is received. Each nonzero value indicates the status of the block. The network function, subfunction, and description are given here. For all values other than 1, the block was not delivered to the user and the task must recover.
		1	Block delivered (FC/ACK).
		2	Block not delivered (FC/NAK/RC=1).
		3	User break 1 (FC/BRK/RC=1).
		4	User break 2 (FC/BRK/RC=2).
		5	Output device not ready (FC/BRK/RC=3).
		6	Incorrectly formatted data (FC/BRK/RC=4).
		7	Stop, terminal busy (FC/STP/RC=1).
		8	Stop, terminal failure (FC/STP/RC=2).
		9	Stop, batch interrupted (FC/STP/RC=3).
		10	Terminal not logged in (CON/END).
		11	Correspondent unavailable or failed, line disconnect (CON/CB/RC=1).
length	Computational-1		Length of the message in characters. This length should include end-of-line bytes for ASCII character types. For display code output, the COBOL compiler generated length is used if the length parameter is omitted.

†Tasks associated with application identifier SY can SEND to any valid TAF user.

TABLE 2-4. FORTRAN SEND REQUEST PARAMETERS

Parameter	Variable Type	Value	Description
mesaddr	Hollerith or left-justified zero fill		Array name containing the message to be sent. For ASCII character types, the message must have an end-of-line byte of the single character US for all logical lines. (Refer to the NAM Reference Manual for NAM data formats.) The TAF SEND interface supplies end-of-line bytes for display code character types only.
length	Integer		Length of the message in characters. This length should include the end-of-line bytes for ASCII character types.
username	Left-justified, zero or blank fill integer		One- to seven-character user name. This must be a name appearing in the NCTFid file, which is validated for the same data base as the task issuing the send. [†] If this parameter is omitted or specified as binary zero, the message is sent to the originating terminal.
cease-flag	Integer	0, omitted	Task end flag. The task continues after the message is sent.
		1	The task ends after the message is sent.
output-flag	Integer		Application block type.
		0, omitted	Indicates that this is the last or only data block in the message.
		1	Indicates that this is not the last data block in the message (if both cease-flag and output-flag are set to 1, there must be another task in the task chain).
block	Integer		Application block number.
		Nonzero	TAF uses the value of this variable as the ABN.
		0	TAF assigns the ABN and returns the value in this variable. For this parameter, the user should enter an integer variable assigned the value 0 instead of entering 0, as follows: INTEGER ABN ABN=0 CALL SEND(mesaddr, length, username, cease-flag,output-flag,ABN,stat)
		Omitted	TAF assigns the ABN, and the value is not returned.
stat	Unnormalized floating point.		Recall flag (applies only if cease-flag is set to 0).
		0, omitted	The task begins processing after the block is transmitted to the network.
		Nonzero	The task begins processing after the block has been delivered to the terminal and acknowledgement is received. Each nonzero value indicates the status of the block. The network function, subfunction, and description are given here. For all values other than 1, the block was not delivered to the terminal and the task must recover.
		1	Block delivered (FC/ACK).
		2	Block not delivered (FC/NAK/RC=1).
		3	User break 1 (FC/BRK/RC=1).
		4	User break 2 (FC/BRK/RC=2).

TABLE 2-4. FORTRAN SEND REQUEST PARAMETERS (Contd)

Parameter	Variable Type	Value	Description
		5	Output device not ready (RC/BRK/RC=3).
		6	Incorrectly formatted data (FC/BRK/RC=4).
		7	Stop, terminal busy (FC/STP/RC=1).
		8	Stop, terminal failure (FC/STP/RC=2).
		9	Stop, batch interrupted (FC/STP/RC=3).
		10	Terminal not logged in (CON/END).
		11	Correspondent unavailable or failed, line disconnect (CON/CB/RC=1).
†Tasks associated with application identifier SY can SEND to any valid TAF user.			

TABLE 2-5. COMPASS SEND REQUEST PARAMETERS

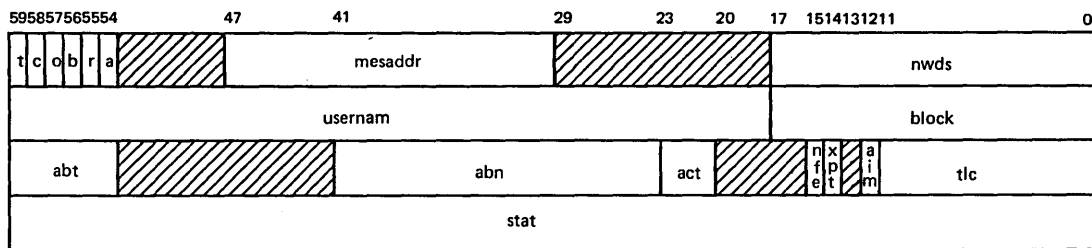
Parameter	Bits	Value	Description
t	59		Terminal flag.
		0	Message is sent to originating terminal.
		1	Message is sent to specified user.
c	58		Task cease flag.
		0	The task cease is not performed.
		1	The task cease is performed following the SEND request.
o	57		Additional SEND requests in message flag. If the a parameter is set to 1, this parameter is ignored and the abt parameter is used.
		0	This is the last or only block in the message.
		1	This is not the last data block in the message.
b	56		Return block flag.
		0	If a is set to 1, the task assigns the block number via the abn field. If a is set to 0, TAF assigns the block number, and the number is not returned in the block field.
		1	TAF assigns the block number and returns this number in the block field.
r	55		Recall flag.
		0	TAF does not wait for the block to be delivered to the terminal.
		1	TAF waits for the block to be delivered to the terminal.

TABLE 2-5. COMPASS SEND REQUEST PARAMETERS (Contd)

Parameter	Bits	Value	Description
a	54	0	Application block header flag. TAF generates the application block header but does it in its own area without making use of or changing word 3 of the parameter table shown in figure 2-8. (Refer to the description of the BLDABH request for the default values TAF uses.) Because a BLDABH request is not available to COMPASS tasks, the header TAF supplies when a=0 always has default values, except for an abt influenced by the output flag (o field) and a tlc influenced by nwds. Specifically, if a is set to 0, the task can only output in 6-bit display code, cannot use format effectors, cannot use auto-input mode, and cannot preset the application block number.
		1	The task supplies the application block header. (The task supplies the ABH in the third word SEND parameter table shown in figure 2-8, with parameters shown in table 2-6.)
mesaddr	47-30		Address of message. For display code character type, the message must contain an end-of-line byte from 12 to 66 bits of zero. For ASCII characters, the message must have an end-of-line byte of the single character US for all logical lines in the output block. (Refer to the NAM Reference Manual.)
nwds	17-0		Number of words in the message. If the a field is set to 1, this field is not used. The tlc field must be used to specify the message length. The nwds field must contain the unit separator.
usernam	59-18		User name from one to seven characters with binary zero fill. If the t field is set to 1, this parameter must be specified. If the t field is set to 0, this field is ignored.
block	17-0		Application block number. If the b field is set to 1, TAF returns the application block number in this field.
abt [†]	59-54		Application block type.
		1	Indicates that this is not the last data block of the message.
abn [†]	41-24	2	Indicates that this is the last or only data block of the message.
			Application block number. If the b field is set to 1, TAF assigns and stores a value in the block field. If the b field is set to 0, this value is stored in block.
act [†]	23-20		Application character type.
		2	8-bit ASCII characters, seven and one-half per word.
		3	8-bit ASCII characters, right-justified in 12-bit bytes, five per word.
		4	6-bit display code characters, 10 per word.
nfe [†]	15		Format effectors.
		0	Format effectors appear in the SEND message. Refer to the NAM Reference Manual for information concerning format effectors.
xpt [†]	14	1	No format effectors appear in the SEND message.
			Transparent bit.
		0	Nontransparent data. IVT transforms are used.
		1	If the act field is set to 2 or 3, IVT transforms are inhibited. If the act field is set to 4, this bit is ignored.

TABLE 2-5. COMPASS SEND REQUEST PARAMETERS (Contd)

Parameter	Bits	Value	Description
aim [†]	12	0	Auto-input mode.
		1	No auto-input mode.
tlc [†]	11-0		Auto-input mode enabled.
			Text length in units specified by the act field. If an output block contains multiple logical lines, all logical lines must contain end-of-line bytes, which must be included in the character count.
stat	59-0		Status of the user name. This field is used only if the r field is set to 1. The value of this field is a supervisor message returned by the network. (Refer to the NAM Reference Manual for a description of the supervisory messages.)
[†] These parameters apply only if the a field is set to 1.			



The parameters are described in table 2-5.

Figure 2-13. SEND Parameter Table

TABLE 2-6. COMPASS SEND PARAMETER INTERRELATIONSHIPS

Field	0	1	0	1	1	Value	0	1	1		
a t b a c t r					0	0	0	=4	4	1	0
Function	Resultant										
MSG terminator (last or only block) or BLK terminator (not the last block) is determined by:	o field	abt field	Originating terminal	Terminal of user usernam	abn field	TAF	TAF				
The message is sent to:											
The block number is assigned by:											
The block number assigned by TAF is:	nwds field	tle field				Not returned	Returned in blk field				
The message length is determined by:											
The character type is determined by:											
Format effector use is determined by:	Only display code can be used	act field									
	Format effectors cannot be used	nfe field									
	Auto-input cannot be used	aim field									
Auto-input use is determined by:											
Transparent mode output is determined by:											
Status of the SEND is returned to:								xpt field	Transparent mode cannot be used	stat field	ITASK and not the sending task

RELSCB Request

The RELSCB request releases all overflow communication blocks[†] to the system. After an RELSCB request is processed, subsequent LOADCB requests return a status word of 0.

The format is:

COBOL

ENTER RELSCB.

There are no parameters.

FORTTRAN

CALL RELSCB

There are no parameters.

COMPASS

RELSCB

There are no parameters.

TSIM Request

The TSIM request provides limited interrogation of the terminal status table. This table, in central memory, is derived from the network file (refer to section 8). The TSIM request can search four fields in the table.

- Data base name field
- User argument field
- Communication line field
- User name field

A list of names of those users which meet a defined criterion is returned to the user. The field specified by code is examined in each terminal table entry by taking the logical product of the field and mask to isolate a portion of the field. The logical difference of this product and the crit value is then obtained. If this result is zero, the terminal entry is placed in list. No action occurs if it is not zero; that is, the user does not meet the specified criterion. For example, an application program could use this to obtain a list of the users which are allowed to use a particular data base by selecting the data base field to be searched for a certain name.

The format is:

COBOL

ENTER TSIM USING code mask crit rleng list leng.

code	Computational-1 item whose value indicates which field is to be searched:
------	---

<u>value</u>	<u>field</u>
--------------	--------------

- | | |
|---|--------------------------|
| 0 | Data base name field |
| 1 | User argument field |
| 2 | Communication line field |
| 3 | User name field |

mask	Parameter whose value is taken as a binary mask. It is used in compiling the list of user names.
crit	Criterion value for the search.
rleng	Computational-1 item whose value TAF sets to indicate the number of entries found.
list	Item in which TAF lists found entries. If zero or absent, no list is returned but rleng is given as the number of found entries.
leng	Computational-1 item whose value indicates the number of words that list can hold. If zero or absent, no list is returned but rleng is given as the number of found entries. Only the lowest order 12 bits of this value are used.

FORTTRAN

CALL TSIM(code,mask,crit,rleng,list,leng)

code	Integer whose value indicates which field is to be searched:
------	--

<u>value</u>	<u>field</u>
--------------	--------------

- | | |
|---|--------------------------|
| 0 | Data base name field |
| 1 | User argument field |
| 2 | Communication line field |
| 3 | User name field |

mask	Parameter whose value is taken as a binary mask. It is used in compiling the list of user names.
crit	Criterion value for the search.
rleng	Integer variable whose value TAF sets to indicate the number of entries found. This value is also returned as a FORTRAN function value.
list	Name of an array in which TAF places a list of found entries. If zero or absent, no list is returned, but rleng is given as the number of found entries.

[†]Described under LOADCB Request.

leng Integer variable whose value indicates the number of words that list can hold. If zero or absent, no list is returned, but rleng is given as the number of found entries. Only the lowest order 12 bits of this value are used.

COMPASS

TSIM addr

addr Address of a parameter table in the format shown in figure 2-14.

TARO Request

The TARO request allows the task to manipulate the user argument field in the terminal status table entry of a particular user. (Refer to section 8 for further information.) The user argument field (24 bits) is operated on in the following manner. The logical product of the user argument field and the lower 24 bits of mask are taken; then the logical difference of this result and value is determined. This value is placed in return and replaces the old user argument field. This allows the user to test a specific field to obtain its value and to establish a particular value in the field. For example, bit 1 of the user argument field may be used to establish a terminal recovery condition. The TARO request operates in the following manner.

Case 1: The request interrogates the user argument field and determines whether the recovery bit (bit 1) is set. The value of this bit in the user area is not changed and is placed in return. (Only bits 0 through 3 are shown in this example.)

logical	010	user argument field of
product	010	TST
		mask parameter
logical	010	
difference	000	value parameter
	010	return value = new user
		argument field

Case 2: The request sets the recovery bit.

logical	000	user argument field
product	010	mask parameter
logical	000	
difference	010	value parameter
	010	return value = new user
		argument field

Case 3: The request clears the recovery bit.

logical	010	user argument field
product	000	mask parameter
logical	000	
difference	000	value parameter
	000	return value = new user
		argument field

	59	47	29	17	0
word 1	code	list	leng	rleng	
word 2	mask				
word 3	crit				

code

Integer whose value indicates which field is to be searched:

value	field
0	Data base name field
1	User argument field
2	Communication line field
3	User name field

list

Location of list to contain found entries. If zero or absent, no list is returned, but rleng is given as the number of found entries.

leng

Number of words that list can hold. If zero or absent, no list is returned, but rleng is given as the number of found entries. Only the lowest order 12 bits of this value are used.

rleng

The address at which TAF returns the number of entries found.

mask

Value to be taken as a binary mask. It is used in compiling the list of user names.

crit

Criterion value for the search.

Figure 2-14. TSIM Parameter Table

The format is:

COBOL

ENTER TARO USING value mask usernam return.

value	Item whose value is a 24-bit value to be used to alter the user argument field.
mask	Parameter whose value is taken as a binary mask. Only the lowest 24 bits are used.
usernাম	A 01-level item containing a one- to seven-character alphanumeric user name. The user name can also be a computational-1 item containing a binary zero (designating the originating user).
return	Item whose value TAF sets to indicate the resultant user argument bits.

FORTRAN

CALL TARO(value,mask,usernam,return)

value	Parameter whose 24-bit value is used to alter user arguments.
mask	Parameter whose value is taken as a binary mask. Only the lowest 24 bits are used.
usernাম	Parameter whose value is a one- to seven-character alphanumeric user name, containing either blank or binary zero fill to a word boundary. If this parameter is omitted or is a binary zero, then the originating user is used.
return	Parameter in which TAF places the resultant user argument bits. This value is also optionally returned as the value of a FORTRAN function.

COMPASS

TARO addr

addr Address of a parameter table in the format shown in figure 2-15.

IIO Request

The IIO request allows a task to change its own I/O limit as measured in RA+1 calls from the task. Any request given in this manual or any data manager request is considered an RA+1 call. The default value is 320 calls.

The format of the request is:

COBOL

ENTER IIO USING nio.

nio Computational-1 item whose value specifies the new I/O limit.

FORTRAN

CALL IIO(nio)

nio Integer whose value specifies the new I/O limit.

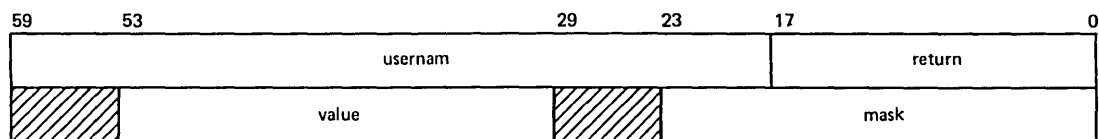
COMPASS

IIO addr

addr Address of a parameter table in the format shown in figure 2-16.

BWAITNP Request (COMPASS Only)

The BWAITNP request allows a task to move data from a terminal to a task field location. This location has the same format as the communication block (refer to figure 2-1). The request is processed the same as a WAITNP request, except that when the terminal data is moved to the specified location, it is moved to the area specified by the supplied parameter (addr), not the first word address of the task (as in the WAITNP request). The first word address is not changed by the execution of this request.



usern	One- to seven-character user name, left-justified with binary zero fill, for which the operation is performed. If usern is a binary zero, then the originating user is used.
return	Location in which to place the resultant user argument bits.
value	24-bit value to be used to alter the user argument field.
mask	Value to be taken as a binary mask. Only the lowest 24 bits are used.

Figure 2-15. TARO Parameter Table



nio New I/O limit.

Figure 2-16. IIO Parameter Table

The format is:

BWAITINP addr

addr Address of first word of the storage area where the terminal data is to be moved.

TAF does not pass this terminal data to tasks called with the CALLTSK and CALLRTN requests.

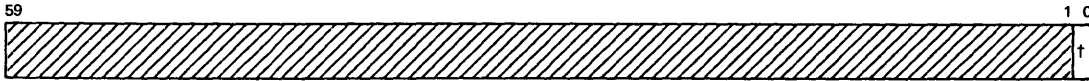
TPSTAT Request (COMPASS Only)

The TPSTAT request allows a task to determine whether it is running under TS or NAM. Terminal control is supported differently, depending on the terminal subsystem. The user can subsequently change the output data to its correct form for the terminal subsystem.

The format is:

TPSTAT addr

addr Address where the identifier of the active terminal subsystem is to be returned as shown in figure 2-17.



<u>t</u>	<u>Meaning</u>
0	TS active (TPTLX)
1	NAM active (TPNAM)

The symbols TPTLX and TPNAM are defined in the assembly text COMKMAC.

Figure 2-17. TPSTAT Return Values

BEGIN Request

The BEGIN request allows a task to specify the area where the terminal input data is to be placed, if an area other than the first word address of the task field length is desired. It supports programming languages which do not reserve the first word address of the task for the common block that is used for the communication block. The task must be installed in the appropriate task library with the (solicited) input directive, because the default is unsolicited. If task chains are involved, the calling task must issue a BEGIN request before resuming execution of the original task, because the calling task must be able to obtain the communication block containing the returned data from the called task. The returned communication block in solicited mode is in the same form as in unsolicited mode. Unless there is a specific need for this request, its use is not recommended because more execution overhead is required to support the secondary load request.

The format is:

COBOL

ENTER BEGIN USING buff.

buff A 01-level item identifying the user-defined buffer area.

FORTRAN

CALL BEGIN(buff)

buff An array name identifying the user-defined buffer area.

COMPASS

BEGIN addr

addr Address of the user-defined buffer area.

TASK CONTROL REQUESTS

These requests are used for task control.

- RECALAL
Pause until all data manager requests are complete.
- CEASE
End a task.
- ITL
Set a task time limit.
- LOGT
Log a terminal out of TAF.
- KPOINT
Initiate execution of K display commands.
- WAIT
Suspend processing for a given amount of time.

RECALAL Request

The RECALAL request causes a pause in task execution until all data manager activity previously initiated for the task is complete. This request is for use with the TAF data manager.

The format is:

COBOL

ENTER RECALAL.

There are no parameters.

FORTRAN

CALL RECALAL

There are no parameters.

COMPASS

RECALAL

There are no parameters.

CEASE Request

The CEASE request provides task end processing and transaction completion processing. If a predetermined task list exists for the chain (that is, subsequent tasks have been requested for scheduling, or the CEASE is a return from a CALLRTN request), no special processing occurs on a CEASE request from a task. However, if the request is from the final task in the chain, the transaction executive releases all communication blocks and notifies the data manager that updated records should be replaced on the data base. The one exception to this procedure occurs if a task selects an abnormal CEASE by use of an optional parameter.

A task which terminates in this manner is automatically the final task in the chain. After an abnormal CEASE occurs, all communication blocks are released and records updated by the task but not yet updated on the data base are not replaced on the data base.

The format is:

COBOL

ENTER CEASE USING a.

a Computational-1 item whose value
 indicates the type of CEASE selected:

<u>a</u>	<u>Type of CEASE</u>
0	Normal CEASE.
1	Abnormal CEASE without memory dump.
Any other	Abnormal CEASE with memory dump.†

FORTRAN

CALL CEASE(a)

a Integer whose value indicates the type of
 CEASE selected:

<u>a</u>	<u>Type of CEASE</u>
0	Normal CEASE.
1	Abnormal CEASE without memory dump.
Any other	Abnormal CEASE with memory dump.†

†Dumps are identified by the transaction subsystem user name or by the appropriate DSDUMP parameter (refer to section 5).

COMPASS

CEASE a

a Integer whose value indicates the type of
 CEASE selected:

<u>a</u>	<u>Type of CEASE</u>
0	Normal CEASE.
1	Abnormal CEASE without memory dump.
Any other	Abnormal CEASE with memory dump.

ITL Request

The ITL request allows a task to set the time limit for the task to a new value. The time limit is expressed in exchange jump time units, that is, the number of times a task executed its full time slice without interruption. In addition, each call to this function decreases the CPU priority of the task by 1 until the lower limit of priority is reached to ensure good service to other tasks. The time limit default value is a product of two installation parameters. The released value is 1.12 CPU seconds. If this is exceeded, the task aborts and the message TIME LIMIT EXCEEDED is issued to the terminal.

The format is:

COBOL

ENTER ITL USING ntm.

ntm Computational-1 item whose value
 specifies the new time limit.

FORTRAN

CALL ITL (ntm)

ntm Integer whose value specifies the new
 time limit.

COMPASS

ITL addr

addr Address of a parameter table in the
 format shown in figure 2-18.

LOGT Request (COMPASS Only)

The LOGT request logs out a terminal from TAF and returns it to nontransaction mode. The terminal can then access other timesharing applications. A sample task is provided using this request. (Refer to System Tasks in section 10.)

The format is:

COMPASS

LOGT

There are no parameters.



ntm New time limit.

Figure 2-18. ITL Parameter Table

KPOINT Request

The KPOINT request allows a task to initiate execution of any K display command. Section 6 describes three K display commands: K.DUMP, K.DUMPLIM, and K.DSDUMP.

K display commands are normally issued from the system console to communicate with and to request specific actions of TAF. K display commands should be issued with care and then only when the results of the command are well-defined.

When a task issues a K display command, the prefix K. is omitted. For example, the entry from the console is K.DUMP., but the same command issued from a task is DUMP..

All K display commands, except K.DUMP, require that the calling task be on the system task library. Any task can issue the K.DUMP command.

The format is:

COBOL

ENTER KPOINT USING addr.

addr 01-level name of the character string (K display command). The length of the character string cannot exceed 80 characters. The string must begin on a word boundary and must be terminated by a period or a right parenthesis; the exception is the K.MESSAGE command, which must be terminated by 6 bits of zeros after either a period or a right parenthesis.

FORTRAN

CALL KPOINT(addr)

addr Address of the character string (K display command). The length of the character string cannot exceed 80 characters. The string must begin on a word boundary and must be terminated by a period or a right parenthesis; the exception is the K.MESSAGE command, which must be terminated by 6 bits of zeros after either a period or a right parenthesis.

COMPASS

KPOINT addr

addr Address of the character string (K display command). The length of the character string cannot exceed 80 characters. The string must begin on a word boundary and must be terminated by a period or a right parenthesis; the exception is the K.MESSAGE command, which must be terminated by 6 bits of zeros after either a period or a right parenthesis.

WAIT Request

The WAIT request provides tasks with the ability to have TAF suspend processing for a specified number of seconds (0 to 3600). The transaction must have no outstanding data manager requests at the time a WAIT request is issued.

The format is:

COBOL

ENTER WAIT USING time.

time A Computational-2 data item which indicates the number of seconds for which processing is to be suspended. If unspecified or out of range, the task is aborted. Only the integer portion is significant.

FORTRAN

CALL WAIT(time)

time A real number or variable which indicates the number of seconds for which processing is to be suspended. If unspecified or out of range, the task is aborted. Only the integer portion is significant.

COMPASS

WAIT addr

addr The address of a word specifying the number of seconds (N) for which processing is to be suspended. The word has the form: 48/0,12/N. If unspecified or out of range, the task is aborted.

TASK UTILITY REQUESTS

The following task utility requests provide capabilities for the FORTRAN and COBOL user, some of which are difficult to do because they are at the bit level.

- **LOGIN**
Determine if a specified user is logged in.
- **EXTRACT**
Extract a bit string from a word.
- **INSERT**
Insert a bit string into a word.

LOGIN Request

The LOGIN request allows a task to determine if a specified user is logged in before it issues a SEND request to that user. TAF aborts a task that issues a SEND request without recall to a user who is not logged in.

NOTE

It is possible for the receiving user to log out after the task issues a LOGIN request and before the task issues a SEND request. Therefore, it is recommended that the task issue the SEND request soon after issuing the LOGIN request.

The format is:

COBOL

ENTER LOGIN USING usernam status.

usern	A one- to seven-character alphanumeric user name, left-justified. A binary zero specifies the user associated with the transaction that is running.
status	Address of the location to contain the returned status. The format of the returned status is computational-1. Values for status are: 0 User not logged in. 1 User logged in.

FORTRAN

CALL LOGIN (usern, status)

usern	A one- to seven-character alphanumeric user name, left-justified. Possible forms are nHf and nLf (refer to the FORTRAN Extended Reference Manual). A binary zero specifies the user associated with the transaction that is running.
-------	--

status Address of the location to contain the returned status. The format of the returned status is integer. Values for status are:

- 0 User not logged in
- 1 User logged in.

EXTRACT Request

The EXTRACT request allows a task to extract a bit string from a word and move it to another word, right-justified. For instance, the EXTRACT request can be used to check fields in the terminal status table or the communication block.

In the following request formats, if n is not within the specified range, if bbb plus 1 minus n is less than zero, or if there are any other errors in the parameters, TAF aborts the task and issues the following message.

TASK REQUEST ARGUMENT ERROR.

The format is:

COBOL

ENTER EXTRACT USING loc1 loc2 bbb n.

loc1	Data name of the source field. Any data type is allowed. loc1 must begin on a word boundary.
loc2	Data name of the computational-1 destination field. The bit string is right-justified in this field.
bbb	Beginning bit position of field loc1; this is the leftmost bit of the field. Bits are numbered 59 through 0.
n	Number of bits to extract; the range of values for n is 1 to 59, inclusive. The format of n must be computational-1.

FORTRAN

CALL EXTRACT (loc1,loc2,bbb,n)

loc1	Name of the integer source field.
loc2	Name of the integer destination field. The bit string is right-justified in this field.
bbb	Beginning bit position of field loc1; this is the leftmost bit of the field. Bits are numbered 59 through 0.
n	Number of bits to extract; the range of values for n is 1 to 59, inclusive. n must be integer type.

The COBOL STRING and UNSTRING statements also provide capabilities for manipulating character-aligned data. The EXTRACT request should be used only when the use of these COBOL statements is impossible or inconvenient.

INSERT Request

The INSERT request allows a task to insert a bit string into a word. For instance, the INSERT request can be used to set fields in the terminal status table.

In the following formats, if *n* is not within the specified range, if *bbp* plus 1 minus *n* is less than zero, or if there are any other errors in the parameters, TAF aborts the task and issues the following message.

TASK REQUEST ARGUMENT ERROR.

The format is:

COBOL

ENTER INSERT USING *loc1 loc2 bbp n*.

- | | |
|-------------|--|
| <i>loc1</i> | Data name of the computational-1 source field. The bit string must be right-justified in this field. |
| <i>loc2</i> | Data name of the destination field. Any data type is allowed. <i>loc2</i> must begin on a word boundary. |
| <i>bbp</i> | Beginning bit position of field <i>loc2</i> ; this is the leftmost bit of the field. Bits are numbered 59 through 0. |

- | | |
|----------|---|
| <i>n</i> | Number of bits to insert; the range of values for <i>n</i> is 1 to 59, inclusive. The format of <i>n</i> must be computational-1. |
|----------|---|

FORTRAN

CALL INSERT (*loc1,loc2,bbp,n*)

- | | |
|-------------|--|
| <i>loc1</i> | Name of the integer source field. The bit string must be right-justified in this field. |
| <i>loc2</i> | Name of the integer destination field. |
| <i>bbp</i> | Beginning bit position of field <i>loc2</i> ; this is the leftmost bit of the field. Bits are numbered 59 through 0. |
| <i>n</i> | Number of bits to insert; the range of values for <i>n</i> is 1 to 59, inclusive. <i>n</i> must be integer type. |

The COBOL STRING and UNSTRING statements also provide capabilities for manipulating character-aligned data. The INSERT request should be used only when the use of these COBOL statements is impossible or inconvenient.

CDCS can be used alone or with one other data manager (CRM, TOTAL, or TAF) in the same transaction. If a task is to use CDCS, the CDCS Reference Manual should be consulted for information on how to set up the data base (schema, subschema, master directory, and files). This data base is associated with CDCS and the files will be at the control point of CDCS during execution. Tasks can use the CDCS verbs for data base access provided that they are compiled with the appropriate subschema relating them to a CDCS data base.

In the following, xx represents the two-character data base identifier.

A TAF application (known by the xx identifier in the TCF) is linked to a CDCS data base by means of the subschemas used in the tasks associated with the application (these are the tasks on xxTASKL). All tasks in a transaction must use the same subschema or a fatal task error (detected by CDCS) will occur.

A transaction will be considered to be connected to CDCS if it has successfully issued a request to CDCS and the terminate request to CDCS has not been completed. (A CDCS terminate request from other than the last task in a chain will be ignored.) If the last task in the chain does not issue a CDCS terminate request, one will be issued automatically when the last task completes.

A task will abort if it attempts to use CDCS when CDCS is not present in the system or if CDCS aborts while a transaction is connected to it. If a transaction aborts while it is connected to CDCS, TAF will issue an abnormal terminate notice to CDCS for the transaction (unless the abort originated with CDCS).

The following TAF files are associated with an application using CDCS:

- TCF } (required)
- xxJ }
- xxTASKL (required unless all tasks are on system task library)
- xxJOR1 } (optional)
- xxJOR2 }
- xxJOR3 }

The possibility of a deadlock involving CDCS and another data manager (TAF, TOTAL, or CRM) exists. For instance, if task X locks some resources under one data manager and then tries to lock other resources under another data manager which task Y initially locked, then the two tasks may be deadlocked if task Y tries to lock any of the resources locked by task X. It is the responsibility of the installation, application writers, and data base administrator to prevent these situations from occurring. Such deadlocks can be prevented by locking resources in a specified order only; for example, lock in data manager A resources first, then data manager B resources, and so forth.

RESTRICT clauses cause tasks to automatically request additional central memory. All of the advice and requirements relevant to a task requesting that its FL be increased apply when RESTRICT clause processing is used (refer to MEMORY Request, section 2).

This section is intended for the data base administrator and can be omitted by the applications programmer.

This section discusses three topics: the xxJ file, journal files, and the TAF configuration file (TCF).

An xxJ file is associated with each data base and has the following functions.

- Defines the user's user name, password, and user index.
- Defines optional journal files.
- Defines optional device residence for the task library and some files specific to the individual data managers (refer to the xxJ file descriptions which follow).

Up to three journal files can be specified optionally to supplement the system journaling. A task can write to one of the journal files by using the JOURNAL request.

TCF, the TAF configuration file, defines the data bases associated with a data manager.

A procedure file (TAFxxxx) is used to initialize TAF. Control statements may be added to it for recovery or to establish the operating environment. For example, control statements which attach trace files (xxTFIL),[†] pool files (xxERPF),[†] journal files (xxJORN), and data base files can be included in this file. Refer to the NOS Installation Handbook for more information (publication number is contained in preface).

xxJ FILE

An indirect access permanent file, xxJ, where xx is the data base name, is required for each data base. If additional journal files are needed, they are defined in the xxJ file. The xxJ file is saved under the transaction subsystem user name. The file is in card image format and varies depending on the data manager used.

NOTE

The transaction subsystem user name should be under the control of a data base administrator. The applications programmer should not have access to this name.

TAF DATA MANAGER xxJ FILE

The following shows the structure of the xxJ file when using the TAF data manager.

```
xxJ
USER(usernum,password,familyname)††
UI=n.
EDT,PN=packname,R=device.
xxJOR1,dv,ft.
xxJOR2,dv,ft.
xxJOR3,dv,ft.
xxTASKL,PN=packname,R=device. } Optional
```

The parameters for the xxJ file are:

```
xx
    Data base name.

n
    User index (obtained from site analyst). This statement is optional.

packname
    Packname of the auxiliary device on which the user's element descriptor table (EDT) or xxTASKL (user's task library) will reside.

device
    Type of device on which the user's file will reside; for example, DI (for 844), or DI3 (for a multiunit 844 consisting of three devices).

dv
    Device type:
        MT    Magnetic tape
        MS    Mass storage

ft
    File type (figure 4-1):
        B     Buffered
        R     Nonbuffered
```

[†]Refer to the TAF/TS Data Manager Version 1 Reference Manual for information on these files.

^{††}Unless an EDT statement is used, TAF locates the data base using the familyname parameter in the procedure file that initializes TAF, and the usernum and password parameters in the xxJ file. The usernum, password, and familyname parameters are used for validation.

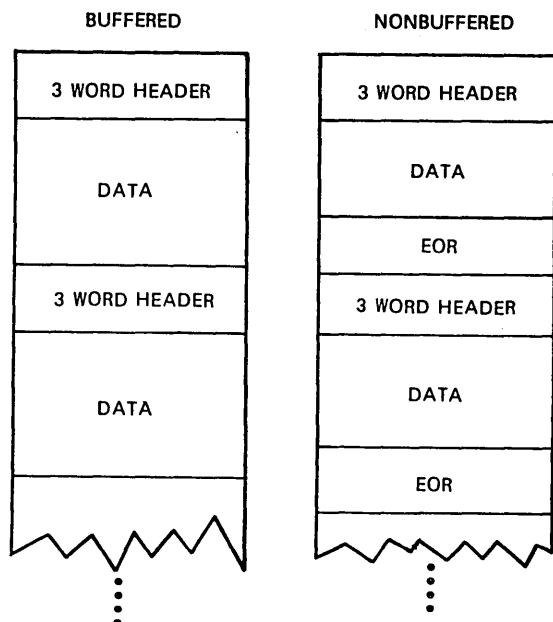


Figure 4-1. Journal File
(Buffered and Nonbuffered Structure)

CRM DATA MANAGER xxJ FILE

The following shows the structure of the xxJ file when using the CRM data manager.

xxJ

USER(username,password,
familyname)[†]

xxJOR1,dt,ft.
xxJOR2,dt,ft.
xxJOR3,dt,ft.
xxTASKL,PN=packname,
R=device.

Optional.

CRM(xxpfnl,type,mode,
users,locks,mrl,kl,
hash,packname,device)

.

.

CRM(xxpfnn,type,mode,
users,locks,mrl,kl,
hash,packname,device)

One CRM statement for
each file.

The parameters are:

xx Data base name.

dt Device type:

MS Mass storage

MT Seven-track tape

NT Nine-track tape

[†] TAF locates the data base using the familyname parameter in the procedure file that initializes TAF, and the username and password parameters in the xxJ file. The username, password, and familyname parameters are used for validation.

ft

File type (figure 4-1):

B Buffered write

R Nonbuffered write

packname Pack name of the auxiliary device on which the user's task library or data base files reside. This parameter is optional on the CRM statement.

device Type of device on which the user's task library or data base files reside. For example, DI (for an 844) or DI3 (for a multiunit 844 consisting of three devices). This parameter is optional on the CRM statement.

xxpfni Two- to seven-character file name. xx is the data base name. pfni is the zero- to five-character data base file name.

type CRM file type:

DA Direct access

IS Indexed sequential

mode Attach mode of file xxpfni:

M Modify, append, or read

R Read

RM Read, with another user concurrently attaching the file in mode M

W Write, modify, append, or read

The mode parameter controls access to the file to ensure that either TAF or a batch job can write on or modify the file at a given time. However, TAF and a batch job can read the file simultaneously. Refer to the ATTACH statement in the NOS Reference Manual, volume 1 for access modes granted for different combinations of multiple access.

NOTE

If the TAF procedure file attaches the file, TAF uses that mode.

users

Number of transactions allowed to open file xxpfni concurrently. When TAF terminates, the TAF dayfile contains a message that gives the number of OPEN requests not granted because the value of the users parameter was exceeded. Each user uses 38 words plus the kl parameter size (in words).

locks	Number of records the transaction can lock for file xxpfni. When TAF terminates, the TAF dayfile contains a message that gives the number of locks not granted because the value of the locks parameter was exceeded. Each lock uses two words plus the kl parameter size (in words).
mrl	Number specifying the maximum record length (in characters) for file xxpfni.
kl	Number specifying the length (in characters) of the key for file xxpfni. [†]
hash	One- to seven-character indirect access file that contains the binary code of the hashing routine for file xxpfni. The entry point for the hashing routine must be this file name (hash). If this parameter is omitted, CRM uses the hashing routine supplied with CRM. This parameter applies only for CRM direct access files.

NOTE

This indirect access file must be stored under the usernum parameter on the USER statement.

TOTAL DATA MANAGER xxJ FILE

The following shows the structure of the xxJ file when using the Total data manager.

xxJ

USER(usernum,password,familyname)^{††}

UI=n. xxJOR1,dv,ft. xxJOR2,dv,ft. xxJOR3,dv,ft. xxTASKL,PN=packname,R=device.	} Optional.
---	-------------

NL. LG,PN=packname,R=device.	} Either the NL statement or the LG statement must be selected. PN and R parameters are optional. NL causes TOTAL to not log recovery data. LG causes TOTAL to log recovery data.
---------------------------------	---

aaaa,PN=packname,R=device . . . zzzz,PN=packname,R=device	} PN and R parameters are optional.
---	-------------------------------------

The parameters for the xxJ file are:

xx	Data base name
n	User index (obtained from site analyst)

dv	Device type:
	MS Mass storage
	MT Seven-track tape
	NT Nine-track tape
ft	File type (refer to figure 4-1):
	B Buffered write
	R Nonbuffered write
packname	Packname of the auxiliary device on which the user's task library (xxTASKL) or data sets will reside
device	Type of device on which the user's file will reside [for example, D1 (for 844) or D13 (for a multiunit 844 consisting of three devices)]
aaaa . . zzzz	} Four-character data set name

CDCS OR NO DATA MANAGER xxJ FILE

If an application uses CDCS and one of the other data managers (TAF, CRM, TOTAL), then the xxJ file should be structured as specified for the other data manager.

If an application accesses only the CDCS data manager or no data manager, the structure of the xxJ file is as follows:

xxJ

USER(usernum,password,familyname)^{††}

xxJOR1,dt,ft. xxJOR2,dt,ft. xxJOR3,dt,ft. xxTASKL,PN=packname,R=device	} Optional
---	------------

The parameters are:

xx	Application (data base) name
dt	Device type:
	MS Mass storage
	MT Seven-track tape
	NT Nine-track tape
ft	File type (figure 4-1)
	B Buffered write
	R Nonbuffered write
packname	Pack name of the auxiliary device on which the user's task library resides
device	Type of device on which the user's task library resides.

[†] The user can specify octal numbers and decimal numbers by using a postradix of B and D, respectively. Decimal is the default.

^{††} TAF locates the data base using the familyname parameter in the procedure file that initializes TAF, and the usernum and password parameters in the xxJ file. The usernum, password, and familyname parameters are used for validation.

CREATING xxJ FILE

To enter an xxJ file, the data base administrator can run a batch job to create and save the file under the transaction subsystem user name. This can also be done at a terminal through IAF, if the transaction subsystem user name is validated to use IAF. (Refer to the IAF Reference Manual for IAF operating instructions.)

JOURNAL FILES

The two types of journal files are system and data base.

SYSTEM JOURNAL FILE

The system journal file (a direct access file named JOUR0 which may be defined under the transaction subsystem user name or may be set up by the TAFxxxx procedure file at initialization) is not associated with any particular data base. All transaction data (input from terminals), errors, and transaction completion notification is automatically recorded on JOUR0. The terminal status table is also journalized periodically. Data sent to terminals by tasks is not recorded on JOUR0. JOUR0 will be created at initialization if not already present under the transaction facility user name.

DATA BASE JOURNAL FILE

The transaction subsystem allows up to three user-specified direct access journal files for each data base. The allowable names are xxJOR1, xxJOR2, and xxJOR3, where xx is the data base name. At initialization, TAF checks the xxJ files associated with all active applications to determine which xxJORN files are needed. It will then attach any specified xxJORN files not already at the control point and attempt to define any which it cannot find. The transaction subsystem user name must have write permission on these files. When a task makes a journal request (refer to JOURNL request), the data base for which the originating terminal is validated is used to determine the journal file on which to write. If the task is associated with a terminal validated for all data bases, the data base specified on the last data base request is used to determine the proper journal file.

JOURNAL FILE ENTRY HEADER

Each journal file entry (refer to JOURNL request) has a header in the format shown in figure 4-2.

59	35	29	17	11	5	0
sequence	org	reserved	length			
task name			h	m	s	
username			reserved			

sequence	Transaction sequence indicator
org	Origin indicator (octal):
	0 Task origin (JOURNL request)
	1 Transaction subsystem origin (input)
	2 Data manager origin
	3 Transaction subsystem recovery/statistical data
	4 End-of-transaction indicator
	5 Incomplete block of terminal input data
	6 Terminal input for an interactive task
	7 Illegal intercontrol point transfer
	10 On-line LIBTASK update (TT option)
	11 CDCS detected error (words following header contain error message)
length	Length of entry (equals header plus data block)
task name	Name of task if task origin requested
h m s	Packed time (hour, minute, second)
username	User name associated with transaction if task origin

Figure 4-2. Journal File Entry Header

JOURNAL REQUEST

The JOURNAL request allows file entries to be made to supplement automatic journalizing.

The format is:

COBOL

ENTER JOURNAL USING jnum message.

jnum Computational-1 item whose value indicates the number of the journal file to which the entries are to be made. The value of this parameter must be right-justified with zero fill.

jnum	Journal File
0	JOUR0 (the system journal file to which the transaction executive normally makes journal entries).
1	Data base journal file xxJOR1.
2	Data base journal file xxJOR2.
3	Data base journal file xxJOR3.

message Name of the 01-level data item containing the message to be journalized.

FORTTRAN

CALL JOURNAL(jnum,message,length)

jnum Integer whose value indicates the number of the journal file to which the entries are to be made.

jnum	Journal File
0	JOUR0 (the system journal file to which the transaction executive normally makes journal entries).
1	Data base journal file xxJOR1.
2	Data base journal file xxJOR2.
3	Data base journal file xxJOR3.

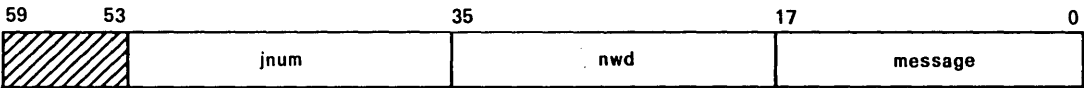
message Name of the array containing the message to be journalized.

length Integer whose value specifies the length of the message in characters.

COMPASS

JOURNAL addr

addr Address of a parameter table in the form shown in figure 4-3.



jnum Journal file number, right-justified with binary zero fill:

0 JOUR0 (the system journal file to which the transaction executive normally makes journal entries).

1 to 3 Data base journal file (xxJOR1, xxJOR2, or xxJOR3).

nwd Length of message in words.

message Address of message to be journalized.

Figure 4-3. JOURNAL Parameter Format

If a journal file is defined on magnetic tape and an end-of-reel is detected, the transaction executive unloads the tape and checks for another tape on which to write. If the operator has not preassigned a unit, the journal file is placed on mass storage. If the operator enters the K.ASSIGN command, the transaction executive copies the mass storage journal file to the tape and places all subsequent entries on the tape. If the mass storage file does not fit on the tape, the message *UNABLE TO USE TAPE* is issued.

TAF CONFIGURATION FILE (TCF)

The TAF Configuration File (TCF) is a required indirect access permanent file saved under the transaction facility user name. It is used to specify which data managers are to be loaded by TAF and which applications (data bases) are to be associated with each data manager. Data in the TCF should be in the format:

DMS,dm,sw,xx1,xx2,...,xxn.
DMS,dm,sw,xx1,xx2,...,xxn.
DMS,dm,sw,xx1,xx2,...,xxn.
DMS,dm,sw,xx1,xx2,...,xxn. } Optional

The parameters on the DMS statements have the following meanings:

Parameter	Meaning
dm	Name of the data manager. Must be TAF, CRM, TOTAL, or OTHER. The same data manager can be specified on any number of DMS statements. The dm value OTHER is to be used for specifying an application which either is not associated with a data manager or is associated only with the CDCS data manager.

Parameter	Meaning
sw	Status. Must be ON or OFF.
ON	The specified data manager (TAF, TOTAL, or CRM) will be loaded. The data bases specified on the same statement are to be made available for TAF processing. These data bases are referred to as active.
OFF	Data bases specified on the same statement are not made available for TAF processing.
xx _i	Data base identifier. Must not be duplicated on the same or any other active DMS statement. For the CRM and TAF data managers, it is the two letter data base name. For the TOTAL data manager, this parameter has the format:

xyyyyy

which is the data base name (the DBMOD file name) specified in the DATA-BASE-NAME statement of the the DBDL statements. xx is the unique TAF data base name and yyyy is any combination of four alphanumeric characters.

A separate DMS statement for CDCS (OTHER) is not necessary if CDCS is to be used in conjunction with another data manager.

The TCF file must contain at least one DMS statement and each DMS statement with status set to ON must have at least one data base identifier or TAF will abort.

The maximum number of data bases which may be associated with one data manager is specified by the installation parameter MAXDB. This value is 25 unless altered by the user's site.

The maximum length of each DMS statement is 160 display code characters.

†Refer to the Total-CDC Version 1 Reference Manual for a detailed description.

The transaction subsystem interfaces with the batch subsystem in two ways. First, batch jobs can initiate tasks by use of the BTRAN request. And second, tasks can route jobs to the batch input queue by use of the SUBMT request.

The BTRAN and SUBMT requests are described in this section.

BTRAN REQUEST

The BTRAN request enables a batch job to start a transaction. The transaction is passed to the transaction executive through the intercontrol point mechanism; therefore, the user must be validated for intercontrol point transfers (AW=CSTP). The user passes a transaction to the transaction executive via intercontrol point area 1 with the header word shown in figure 5-1.

The format is:

COBOL

ENTER BTRAN USING tran status.

tran	Name of the 01-level item containing the header word for the transaction to be submitted.
status	Name of the data that contains the returned status; the status is returned as a computational-1 value.

Octal status values are as follows:

- 1 Transaction submission is complete.
- 3 The transaction executive is not up.
- 7 Transaction data is too long; maximum length is 620 characters or less according to installation option.

- 11 This user cannot submit batch jobs to TAF. Bit 9 (CSTP) in the user's access word is not set. Refer to the MODVAL control statement in the System Maintenance Reference Manual for further information.

If the user name is invalid, the transaction executive sends the following message to the TAF dayfile.

ILLEGAL TERMINAL NAME

This message appears at the system console. Users receiving questionable results should contact the console operator to see if this message appeared while they were trying to submit a batch transaction.

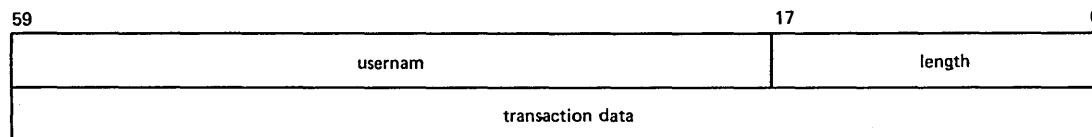
The EXTRACT and INSERT requests (section 2), used in conjunction with computational-4 items, can be used to construct the data block necessary for this request (figure 5-1).

FORTRAN

The user calls the BTRAN request in two ways. The first method is a FORTRAN function.

i=BTRAN(tran)

i	Integer variable that contains the request status after BTRAN is called. The user must declare BTRAN as an integer.
tran	Name of the array containing the header word for the transaction to be submitted.



username	One- to seven-character user name, left-justified with zero fill, for which the operation is to be done.
length	Length of the transaction in words, includes the header word and the transaction data.
transaction data	Transaction name (or transaction name and input data) to be processed by TAF.

Figure 5-1. BTRAN Header Word

The second method is a subroutine call.

CALL BTRAN(tran,status)

tran Name of the array containing the header word for the transaction to be submitted.

status Address of the variable that contains the returned status; the status is returned as an octal integer value.

Octal status values are:

- 1 Transaction submission is completed.
- 3 The transaction executive is not up.
- 7 Transaction data is too long; maximum length is 620 characters or less according to installation option.
- 11 This user cannot submit batch jobs to TAF. Bit 9 (CSTP) in the user's access word is not set. Refer to the MODVAL control statement in the System Maintenance Reference Manual for further information.

If the user name is invalid, the transaction executive sends the following message to the TAF dayfile.

ILLEGAL TERMINAL NAME

This message appears at the system console. Users receiving questionable results should contact the console operator to see if this message appeared while they were trying to submit a batch transaction.

```
BTRANEX.
USER,XYZ1234,XYZ1.
CHARGE,1234,567Z890.
COBOL5.
LDSET(LIB=BDMLIB)
LOAD,LGO.
NOGO,LGOB.
LGOB.
- -EOR- -
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. BTRANEX.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. CYBER.
OBJECT-COMPUTER. CYBER.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 BATCH-TRANSACTION.
   02 TERMINAL-NAME            VALUE IS "ABC1234"        PIC X(7).
   02 DATA-LENGTH            VALUE 2        COMP-4        PIC 9999.
   02 TRANSACTION-DATA        VALUE IS "EX.M12"        PIC X(6).
01 BTRAN-STATUS                VALUE IS 0        COMP-1        PIC 9999.
PROCEDURE DIVISION.
START-RUN.
```

```
ENTER BTRAN USING BATCH-TRANSACTION, BTRAN-STATUS.
DISPLAY "STATUS=" BTRAN-STATUS.
STOP RUN.
```

COMPASS

BTRAN tran

tran Address of the header word for the transaction to be submitted.

The status of the request is returned in register X6.

Octal status values are as follows:

- 1 Transaction submission is complete.
- 3 The transaction executive is not up.
- 5 The transaction executive is busy because the batch buffer is full, the subsystem is being moved, or the subsystem is job-advancing. Resubmit the request.
- 7 Transaction data is too long; maximum length is 620 characters or less according to installation option.
- 11 This user cannot submit batch jobs to TAF. Bit 9 (CSTP) in the user's access word is not set. Refer to the MODVAL control statement in the System Maintenance Reference Manual for further information.

If the user name is invalid, the transaction executive sends the following message to the TAF dayfile.

ILLEGAL TERMINAL NAME

Figure 5-2 demonstrates the use of BTRAN.

Figure 5-2. BTRAN Request

M12 is the name of the transaction to be processed by TAF. Output from SEND or DISPLAY statements in M12 will appear at terminal ABC1234, if it is logged in to TAF, unless otherwise directed within M12. STATUS= n goes to the OUTPUT file associated with the batch job.

SUBMT REQUEST

The SUBMT request allows a task to route a job to the local batch input queue. The address parameter in the call points to the area in the task field length that contains the job to be routed in control word format. This format is described in conjunction with the READCW macro in the NOS Reference Manual, volume 2 (refer to preface for publication number). Using this format, the job may consist of multiple records.

The format is:

COBOL

ENTER SUBMT USING addr.

addr Name of the 01-level item containing the job data to be routed. This data is in the format shown in figure 5-3.

FORTTRAN

CALL SUBMT (addr)

addr Name of an array containing the job data to be routed. This data is in the format shown in figure 5-3.

COMPASS

SUBMT addr

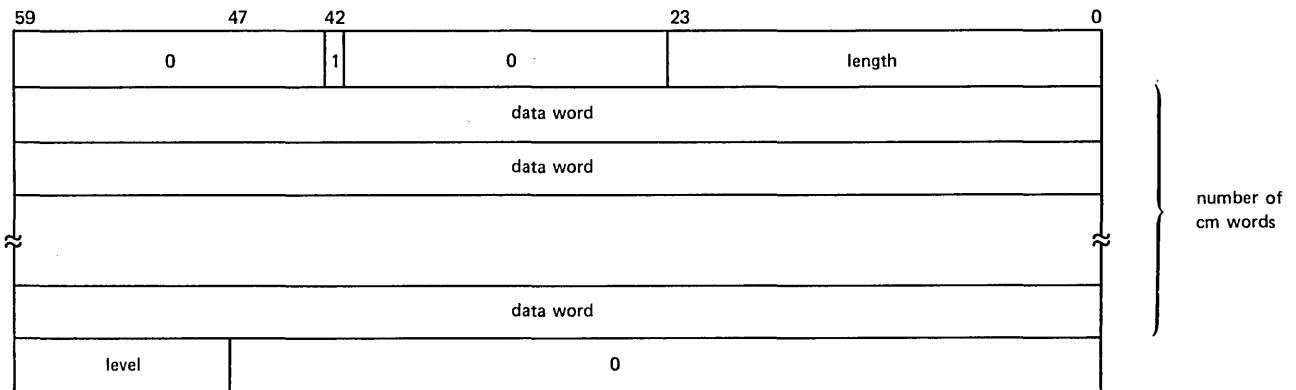
addr Address of the first word of the job data to be routed in control word format as shown in figure 5-3.

The first word at addr must be a control word in the format shown previously. The length field must specify the number of 12-bit bytes occupied by the data in the PRU (maximum of 320 bytes, equivalent to 640 characters). The level number in the trailing control word specifies whether the PRU is an end-of-record or end-of-file. If a logical record extends over more than one PRU, the level number for the first PRU of the record must be nonzero and other than 17₈.

A leading and a trailing control word must be present for each PRU of data.

Typically, the data would consist of control statements which would comprise one record. Additional data to be processed by the batch job could also be present and typically constitute subsequent records.

Figure 5-4 shows a task using the SUBMT request.



length Length in 12-bit (2-character) bytes of job data following this control word (must be five times the number of central memory words).

level Logical record level number:

0 or omitted This physical record unit (PRU) is an end-of-record.

17₈ This PRU is an end-of-file.

Figure 5-3. SUBMT Parameter Table

```

00100 IDENTIFICATION DIVISION.
00110 PROGRAM-ID. SUBTEX1.
00120 ENVIRONMENT DIVISION.
00130 DATA DIVISION.
00140 COMMON-STORAGE SECTION.

```

Standard COBOL Communication Block (refer to section 2)

```

00370 WORKING-STORAGE SECTION.
00380 01 JOB.
00390 03 FILLER VALUE IS 1 COMP-4 PIC 9(5).
00400 03 FILLER VALUE IS 0 COMP-4 PIC 9(5).
00410 03 JOB-LENGTH VALUE IS 75 COMP-4 PIC 9(6).
*
* THE FOLLOWING IS THE JOB TO BE SUBMITTED
*
00420 03 JOB-CARD.
00430 05 JOB-ID VALUE IS "JOB." PIC X(4).
00440 05 FILLER VALUE IS ALL ":" PIC X(6).
00450 03 JOB-USER.
00460 05 USER-CARD VALUE IS "USER,DWG2603,ABC123." PIC X(20).
00470 05 FILLER VALUE IS ALL ":" PIC X(10).
00480 03 JOB-CHARGE.
00490 05 CHG-CARD VALUE IS "CHARGE,5927,693X451." PIC X(20).
00500 05 FILLER VALUE IS ALL ":" PIC X(10).
00510 03 JOB-DATA.
00520 05 COMMENT VALUE IS "* SUBMIT BATCH JOB." PIC X(19).
00530 05 FILLER VALUE IS ALL ":" PIC X(11).
00540 05 CTR-STAT-1 VALUE IS "REWIND,INPUT." PIC X(13).
00550 05 FILLER VALUE IS ALL ":" PIC X(7).
00560 05 CTR-STAT-2 VALUE IS "COPYSBF." PIC X(8).
00570 05 FILLER VALUE IS ALL ":" PIC X(2).
00580 05 CTR-STAT-3 VALUE IS "ROUTE,OUTPUT." PIC X(13).
00590 05 FILLER VALUE IS ALL ":" PIC X(7).
*
* SUBMITTED JOB ENDS HERE
*
00600 03 INPUT-LEVEL VALUE IS 15 COMP-4 PIC 99.
00610 03 FILLER VALUE IS ALL ":" PIC X(8).
00620 PROCEDURE DIVISION.
00630 START-UP.
00640 ENTER SUBMT USING JOB.
00650 DISPLAY "JOB SUBMITTED.".
00660 STOP RUN.

```

Figure 5-4. SUBMT Request

The first word of JOB (00390-00410) sets bit 42 in the header word and gives the length (75) in 12 bit bytes of the card deck images to follow. This is 5 times the number of 10 character (60 bit) words in the job being submitted, in this case, statements 00420-00590.

Each job control statement must be from 1 to 80 characters long. All control statements to be submitted

must be padded to a multiple of 10 characters and the last 2 characters must be binary zero (character ":" in the 64-character set). Thus, if a statement ends in character position 9,10; 19,20; and so on, a zero word must be appended. Statements 00460-00530 give examples of this.

A ROUTE statement (00580) is necessary to produce printed output.

A task can perform two types of dumps. The first type is a task dump, a dump of the task field length, exchange package, and associated data buffers. A task dump aids the applications programmer in debugging a task. The second type is a TAF dump, a dump of all or part of the TAF field length. A TAF dump contains pertinent information that is not contained in a task dump. This section describes the methods used to obtain both types of dumps.

TASK DUMP

The central memory dump (CMDUMP) request dumps central memory according to the parameters specified in the direct subsequent dumps (DSDUMP) request. Parameters specified on the DSDUMP request are used under three circumstances.

- When a fatal task error occurs.
- When a default value is requested in a CMDUMP request.
- When a parameter is in error in a CMDUMP request.

The purpose of DSDUMP and CMDUMP is to aid application programmers in debugging their tasks. These dump requests allow users to dump any portion of their programs. The user specifies the first word address (fwa) and the last word address (lwa) of the area to be dumped. The exchange package and/or all currently held TAF data manager data base file buffers may also be dumped by setting the proper parameter(s) to a nonzero value. With the CMDUMP command, users can specify particular TAF data manager file buffer areas in memory which are to be dumped.

The DSDUMP request sets up default values for subsequent CMDUMP requests made by this task and all linked tasks in this transaction (those called by CALLTSK without CEASE). DSDUMP does not cause a dump; it only sets parameter values that can be defaulted as a result of a CMDUMP request.

A hierarchy of parameter values allows users to default any parameters they want on DSDUMP or CMDUMP calls. At system assembly time, default values are assembled for each parameter. These default values can be changed via a K display command. In addition, if users want to change any default parameter values for their tasks, they may do so by using DSDUMP requests.

This hierarchy of default values works in the following manner. The CMDUMP request causes a dump according to

the parameters it specifies. If a default option is requested, the system determines if a DSDUMP request has been made for this transaction. If a DSDUMP request has been made and if the corresponding parameter value has been supplied, this value is used. If DSDUMP requests have not been specified, a new default value, the system default value for this parameter, is used.

All task dumps, whether due to an aborted task, an abnormal CEASE request, or a CMDUMP request, are handled in the same manner. A job file is created containing the information to be dumped in binary format, as well as a control statement record. The transaction executive issues a ROUTE request releasing the job file to the input queue for batch processing. A program called KTSMDMP is provided, either to generate listable output from the binary file or to transfer it to a direct access permanent file.

When dumping to an output queue, a ROUTE statement is included in the control statement record to direct the output produced by KTSMDMP to the proper queue.

A USER statement is always included, thus enabling KTSMDMP to attach the appropriate file if the dump is destined for a user's file. In this case, KTSMDMP appends the binary information to the end of the permanent file. This method of handling dumps relieves the transaction executive of time-consuming processing that is better performed by a utility.

DSDUMP REQUEST

The DSDUMP request allows the application programmer to change any of the default values of the CMDUMP request. The DSDUMP does not cause a dump in itself, except when the system detects an error (that is, aborts). Any default values on the DSDUMP are satisfied from the general default values. All six parameters are required. If an error occurs on any DSDUMP parameter, the task is aborted.

The format is:

COBOL

ENTER DSDUMP USING fwa lwa ep db oq qd.

fwa	Computational-1 item which specifies beginning address of task memory to be dumped; legal values are $0 \leq fwa \leq FL$ (field length).
-----	---

If fwa is negative, the default value is used.

lwa Computational-1 item which specifies last word of task memory to be dumped; it must be greater than fwa. If 0 is specified, no dump of task memory occurs. If lwa equals FL, the dump extends to the end of the field length. If lwa is negative, the default value is used.

ep Computational-1 item which specifies exchange package option.

0 Do not dump exchange package.

> 0 Dump exchange package.

< 0 Use default value.

db Computational-1 item which specifies TAF data manager buffer option (for use with TAF data manager only).

0 Do not dump TAF data manager buffers.

> 0 Dump TAF data manager buffers.

< 0 Use default value.

oq Computational-1 item which specifies output queue option.

< 0 Use default value (see note below).

0 Dump to local batch output queue.

1 Dump to remote batch output queue.

2 Dump to user permanent file defined under data base user number.

qd Queue destination option; value depends on oq selection.

oq qd

0 Printer ID, specified as a comp-1 number. If negative, use default value (see note below).

1	User number	A valid name specified in display-coded, left-justified characters with either blank or binary zero fill.
2	Direct access permanent file name	

NOTE

If either oq or qd is negative, the default value will be used for both, regardless of the option selected for the other.

FORTRAN

CALL DSDUMP(fwa,lwa,ep,db,oq,qd)

fwa Octal integer which specifies the beginning address of task memory to be dumped; legal values are 0 fwa FL (field length).

If fwa is negative, the default value is used.

lwa Octal integer which specifies the last word of task memory to be dumped; must be greater than fwa. If 0 is specified, no dump of task memory occurs. If lwa equals FL, the dump extends to the end of the field length. If lwa is negative, the default value is used.

ep Integer which specifies the exchange package option.

0 Do not dump exchange package.

> 0 Dump exchange package.

< 0 Use default value.

db Integer which specifies TAF data manager buffer option (for use with TAF data manager only).

0 Do not dump TAF data manager buffers.

> 0 Dump TAF data manager buffers.

< 0 Use default value.

oq Output queue option.

< 0 Use default value (see note below).

0 Dump to local batch output queue.

1 Dump to remote batch queue.

2 Dump to user permanent file defined under data base user number.

qd Queue destination option; value depends on oq selection.

oq qd

0 Integer specifying printer ID. If negative, use default value (refer to note below).

1	User number	A valid name specified in display-coded, left-justified characters with either blank or binary zero fill.
2	Direct access permanent file name	

NOTE

If either oq or qd is negative, the default value will be used for both, regardless of the option selected for the other.

COMPASS

DSDUMP addr

addr Address of a parameter table. The table has the form shown in figure 6-1.

CMDUMP REQUEST

The CMDUMP request allows the application programmer to dump memory in the task's field length and/or data manager buffers to a selected output queue (and logical device). The first six parameters are required. If there is an error on or a default of any parameter specified by CMDUMP, the parameter value of the DSDUMP call is used.

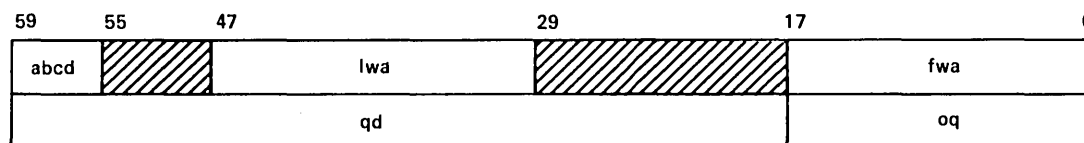
The format is:

COBOL

ENTER CMDUMP USING fwa lwa ep db oq qd file₁ file₂...file_n.

The descriptions of the first six parameters are the same as those for the DSDUMP request.

file_i File name of a data manager buffer to be dumped. Legal values for file_i are four display-code characters, left-justified. If no file names are present, a dump of all data manager buffers is implied.



abcd Bits 59 through 56 of first COMPASS parameter word.

59 Set to 1 for exchange package dump.

58 Set to 1 for TAF data manager buffer dump.

57 Set to 1 for exchange package default value.

56 Set to 1 for TAF data manager buffer default value.

lwa Last word of task memory to be dumped; must be greater than fwa. If 0 is specified, no dump of task memory occurs. If lwa equals FL, the dump extends to the end of the field length. If lwa is negative, the default value is used.[†]

fwa Beginning address of task memory to be dumped; legal values are 0 fwa FL (field length).

If fwa is negative, the default value is used.[†]

qd Queue destination option; value depends on oq selection.

oq

qd

0 Printer ID, specified as a binary number right-justified within word; if negative, use default value.

1 User name

2 Direct access
 permanent file name

} A valid name specified in display-coded,
 left-justified characters with either blank
 or binary zero fill.

oq Output queue option.

0 Dump to local batch output queue.

1 Dump to remote batch output queue.

2 Dump to user permanent file defined under data base user name.

Figure 6-1. DSDUMP Parameter Table

[†]Default values are specified by setting the sign bit.

FORTRAN

CALL CMDUMP(fwa,lwa,ep,db,oq,qd,file₁,
file₂,...,file_n)

The descriptions of the first six parameters are the same as those for the DSDUMP request.

file_i File name of a data manager buffer to be dumped. Legal values for file_i are four display-code characters, left-justified. If no file names are present, a dump of all data manager buffers is implied.

COMPASS

CMDUMP addr

addr Address of a parameter table in the form shown in figure 6-2.

NOTE

The manner in which the hierarchy of default values works was previously described. However, in the special case of output queue and queue destination (oq and qd parameters), if either parameter is defaulted, the next pair of output queue

and queue destination parameters is obtained from the next level of the default hierarchy.

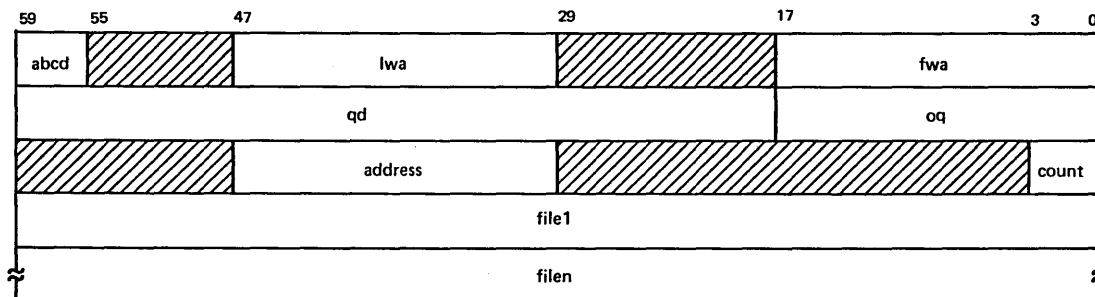
There also is a special case concerning errors in a CMDUMP request. If the parameter in error is the output queue value, the system is unable to route the dump as the user wants. If this error is on a CMDUMP request, the system attempts to dump according to the corresponding DSDUMP output queue and queue destination parameters. If that is not possible, the general system values are used.

K.DSDUMP COMMAND

The K.DSDUMP command can be used under the K display for the transaction executive control point to reset default dump parameters. The parameters can be given in any order and only those parameters to be changed need to be given. If the data base option is specified, all TAF data manager file buffers currently belonging to a transaction are dumped when a dump defaults to this parameter. If (DB)=0, no buffers are dumped.

The format when entered from the console is:

K.DSDUMP,FW=fwa,LW=lwa,EP=ep,OQ=oq,
QD=qd,DB=db.



The descriptions of the first five parameters are the same as those shown in figure 6-1.

address User's calling address to be placed on listing.

count Number of file names present.

file_i File name of a TAF data manager buffer to be dumped. Legal values for file_i are four display-code characters left-justified and binary zero filled. If no file names are present, a dump of all data manager buffers is implied.

Figure 6-2. CMDUMP Parameter Table

The values of these parameters are as specified for the DSDUMP request made by a task. These system dump parameters can be examined by checking the K display for control points.

KTSDMP UTILITY

KTSDMP can be called via a control statement to form listable memory dumps. The input files can be created by the transaction executive or by any other program. The following items can be listed.

- Central memory of a task or program.
- Exchange package and control point area of a task.
- Communication block of a task (always present).
- TAF data manager data buffers.

The format is:

KTSDMP (lfn₁,lfn₂,P,O)

lfn ₁	Input file name; if absent, default is INPUT.
lfn ₂	Output file name; if absent, default is OUTPUT.
P	If P is specified, append file lfn ₁ on direct-access permanent file lfn ₂ . The job must be running with the proper user name and file lfn ₂ must be defined. End-of-file is not copied to lfn ₂ .
O	If O is specified, the dump is in octal format. If O is omitted, the dump is in octal and display code format, with the display code format to the right of the octal format.

If the P option is used, an installation may keep a running stack of memory dumps to be listed selectively at some later time. To list all dumps on the permanent dump file, the file should be attached and KTSDMP executed. All dump records on the file are processed. To list only selected dumps, the permanent file should be attached and the selected dump records extracted using GTR (refer to the NOS Reference Manual, volume 1). KTSDMP should be executed to process the output file from the GTR request.

The user must specify the O option when issuing the KTSDMP control statement from a terminal; if the O option is not specified, execution is terminated, and KTSDMP issues the following message.

DISPLAY DUMP NOT ALLOWED TO TERMINAL.

MEMORY DUMP FORMATS

Figure 6-3 contains the formats of the dumps provided. The leftmost date and time in the header are the date and time of creation of the KTSDMP input file. The current date and time follow. The task control point area dump appears only when the exchange package dump is requested. The exchange package dump, task control point

area dump, and the communication block dump all appear on the same page when the exchange package dump is requested.

TAF DUMP

A TAF dump is a dump of all or part of the TAF field length. The user initiates a TAF dump using the K.DUMP command. A TAF dump assists the analyst in detecting system errors which do not abort TAF. When a task detects an error in TAF, a TAF dump might be needed because there is significant information outside the task field length that is not contained in a task dump.

A task initiates a TAF dump using the KPOINT request (refer to KPOINT request in section 2) in conjunction with the K.DUMP command. The K.DUMPLIM command sets a global limit on the number of K.DUMP commands that tasks can issue. A task uses the KPOINT request in conjunction with the K.DUMPLIM command to set the global task dump limit (GTDL). The remainder of this section describes the K.DUMP and K.DUMPLIM commands.

K.DUMP COMMAND

The K.DUMP command initiates a dump of all or part of TAF. The user can specify the first word address (fwa) and the last word address (lwa) of the memory to be dumped.

The command formats when issued from a task are:

DUMP,fwa,lwa.

or

DUMP,lwa.

or

DUMP.

fwa Octal address of the first word of memory to be dumped; legal values are $0 \leq fwa \leq lwa$. The default is zero. This parameter cannot be used unless the lwa parameter is also used.

lwa Octal address of the last word of memory to be dumped; legal values are $fwa \leq lwa \leq 377777$. The default is 377777.

If both parameters are omitted, the entire field length is dumped. The output of the dump is routed to a printer which has an identifier of zero. A header page precedes the dump; it lists the date and time of the dump and the name of the task that issued the dump. Also listed are the instructions that the dump be given to the TAF central site systems analyst.

K. DUMPLIM COMMAND

The K.DUMPLIM command sets the GTDL. The GTDL specifies the number of K.DUMP commands that tasks can issue.

```

KTSDMP - tasknam          yy/mm/dd. hh.mm.ss.          yy/mm/dd.hh.mm.ss.
EXCHANGE PACKAGE.
    Exchange package dump
TASK CONTROL POINT AREA.
    Task control point area dump
COMMUNICATION BLOCK DUMP FROM fwa TO lwa
    PROGRAM tasknam  SEQUENCE num  ADDRESS xxxxx
    Communication block dump
KTSDMP - tasknam          yy/mm/dd. hh.mm.ss.          yy/mm/dd. hh.mm.ss.
DUMP FROM fwa TO lwa
    Memory dump
KTSDMP - tasknam          yy/mm/dd. hh.mm.ss.          yy/mm/dd. hh.mm.ss.
DUMP FROM fwa TO lwa
    Memory dump
KTSDMP - tasknam          yy/mm/dd. hh.mm.ss.          yy/mm/dd. hh.mm.ss.
DATA BUFFER DUMP OF db filen
KEY VALUE
    Value
PRU          xxx
STATUS      xxx
    File buffer dump

```

Figure 6-3. KTSDMP Format

The command format when issued from a task is:

DUMPLIM,n.

n The decimal number of TAF dumps that tasks can issue. The default is zero. The maximum value is 9999999. The user specifies octal by using a postradix of B.

The initial value for the GTDL is zero. When the GTDL is zero, no TAF dumps from tasks are allowed. The GTDL must be set to a value greater than zero for TAF dumps from tasks to occur. Each TAF dump (where the task does not abort) causes the GTDL to be decreased by 1. This command does not affect the execution of task dumps.

This section can be omitted by the applications programmer.

RECOVERY

Transaction control-point recovery occurs under the following conditions when sense switch 4 is set.

- If the transaction subsystem is aborted for any reason.
- If the entire operating system is recovered (level 3 system deadstart recovery).

The sense switch is set by the operator. (Refer to the n.ONSX command in the NOS Operator's Guide.) If automatic recovery is not desired, sense switch 4 should not be set.

Recovery is automatic, requiring no operator interaction. The following dayfile messages are issued during recovery.

```

RECOVERY IN PROGRESS.
n.nnn KILO TRANSACTIONS PROCESSED.
n.nnn KILO TRANSACTION ABORTS.
n.nnn KILO TASK RELOADS.
n.nnn KILO ITASK RELOADS.
n.nnn KILO TAF FL INCREASES.
n.nnn KILO STORAGE MOVES OF TASKS.
n.nnn KILO NO FL FOR TASK LOAD.
n.nnn KILO NO SUBCONTROL POINTS.
n.nnn KILO NO COMMUNICATION BLOCKS.
n.nnn KILO TASK ROLLOUT STARTS.
n.nnn KILO TASK ROLLOUT COMPLETES.
n.nnn KILO TASK ROLLOUTS BY TAF DM.
n.nnn KILO TASK STARTS BY TAF DM.
n.nnn KILO TASK RECALLS.
n.nnn AVERAGE ACTIVE SUBCONTROL POINTS.
n.nnn AVERAGE OUTSTANDING CDCS REQUESTS.
n.nnn KILO CDCS REQUEST REJECTS FOR MAXR.
n.nnn KILO CDCS REQUEST REJECTS FOR BUSY.
n.nnn KILO CDCS REQUESTS FROM TASKS.
n.nnn PER CENT CPU USAGE.
RECOVERY COMPLETE.

```

The following dayfile messages are issued during recovery only if the CRM data manager is being used.

```

n.nnn KILO OPENS - filename.
n.nnn KILO OPEN REJECTS - filename.
n.nnn KILO LOCKS - filename.
n.nnn KILO LOCK REJECTS - filename.
AAM FILES CLOSED.

```

The filename field is the name of the CRM permanent file.

The transaction abort count is the number of times a transaction did not end normally due to a transaction-executive-detected error in a task while processing the transaction. Errors of this type include mode errors (in the task), illegal parameters on a request from a task, and task CPU time limit exceeded (possible infinite loop in task). This count does not include transactions terminated abnormally by use of the CEASE request.

If sense switch 4 was not set, the preceding messages are still recorded but the transaction executive TERMINATE message is issued instead of the RECOVERY COMPLETE message. If sense switch 5 is set, the transaction executive dumps its field length and releases the output file to the local print queue after each fatal error or operator drop.

If recovery is attempted before initialization is complete, the following messages appear.

```

RECOVERY IN PROGRESS.
RECOVERY IMPOSSIBLE.

```

NOTE

The procedure file used to initialize TAF must include the proper control statements to recognize sense switches and perform dumps and recovery. Refer to the NOS Installation Handbook for further information.

The first action taken during recovery is to flush buffered journal files. The initialization overlay is then called to reload the transaction subsystem data manager code and build memory resident tables, such as the terminal status table and task library directories. After the tables are constructed, the recovery flag (word 0 of the communication block) is set in the user area of each terminal status table entry to enable initial task or any other task to detect that a recovery has taken place. The location of the flag in the area may be established by each installation. The flag is accessible by the use of either the TARO or TSIM request. The flag can be turned off by the TARO request.

The rebuilding of memory resident tables nullifies any changes made by K display commands prior to the recovery. It is the responsibility of the operations personnel, after a recovery has occurred, to enter desired changes, if any. Also nullified are any changes made to the user areas of the terminal status table by the TARO request. It is the responsibility of the applications programmer to reset or clear the bits that were changed prior to recovery.

No transactions are recovered; thus, it is the user's (application task's) responsibility to recover from partially completed transactions and their effects on the data base.

A K display command K.REC=YES. can be used to set terminal status table recovery bits during a normal initialization. This could be used after a 0 level deadstart to force the terminals into recovery mode.

Under the following conditions, the transaction executive sets the recovery bit in the user area of the terminal status table.

- If the K display command K.REC=YES. is entered at initialization time, the transaction executive sets the recovery bit for all terminals.
- If NAM aborts or immediate shutdown occurs while the transaction executive is running, the transaction executive clears the log-in bit and sets the recovery bit for all users currently logged into the transaction subsystem. If any tasks are rolled out waiting for terminal input, the transaction executive sets their rollout time delays to zero such that they are rolled in.
- If a transaction terminal user hangs up the phone or a line problem occurs, the transaction executive clears the log-in bit and sets the

recovery bit for that user name. Clearing the log-in bit enables the terminal user to reenter transaction mode.

Any of these conditions which cause the transaction executive to set the recovery bit indicates a possible loss of transaction terminal output. The application task monitors the recovery bit in the user area field and takes appropriate terminal/application resynchronization procedures when the bit is set.

When certain network abnormal conditions occur, TAF-originated transactions notify ITASK about recovery situations. For more information, refer to section 10. The block parameter on the SEND request for TAF allows the application to identify messages on a file. Upon receiving NAM break messages, the transaction executive creates a TAF-originated transaction which contains information on the break reason and the block on the affected SEND.

The stat parameter on the SEND request allows examination of error conditions by the task which initiated the SEND. If the stat parameter is used, ITASK is not notified of error conditions.

OPERATOR INTERFACE

A console K display is provided to display transaction subsystem status at all times. A set of console operator commands is also available for changing certain subsystem parameters. The NOS Operator's Guide supplies a description of the commands (refer to preface for publication number).

The applications programmer should determine whether an analyst is preparing the network files. If so, this section can be omitted.

Before any transaction processing can be done, the terminal configuration must be defined to the transaction executive. This is done by the terminal network description file (network file).

The NAM local configuration file is also needed to describe the terminal configuration. This file is described in the Network Definition Language Reference Manual.

TERMINAL NETWORK DESCRIPTION FILE

A terminal network description file consists of directives that describe users and their data bases. The directives needed consist of the //DIAL directive and user directives.

The format is:

```
//DIAL.
```

User directives must follow the //DIAL directive. User directives have the form:

```
/username,parameter-string.
```

username	A user name which is used to log into the network and TAF (or is specified in the local configuration file for automatic login).
----------	--

The possible elements of the parameter string, separated by commas, are:

TT=*ID	Transaction terminal type.
DB=aa	Two-character name of the data base to be accessed by the user. Only one data base can be associated with a user name.
RS=n	The data manager read security for the user, with respect to data base, is specified by RS parameter. The user cannot access data elements whose DATADEF read security is higher than this value. n ranges from 0 through 7 with 7 the most exclusive. Default value is zero.
US=n	This declares the data manager update security for the user. The user cannot update elements with update security higher than this value. n ranges from 0 through 7 with 7 the most exclusive. Default value is zero.
UA=n	Contents of the user argument area in the communication block. n ranges from 0 through 16777215. Default is zero. User area contents can be examined and changed by tasks. The appropriate initial value is site dependent. Bit zero is the default task recovery bit.

A simple set of terminal network file directives follows:

```
//DIAL.
/AJOHNSN,TT=*ID,RS=7,US=3,DB=BC,
  UA=16117711B.
/USERNUM,TT=*ID,RS=6,US=7,DB=BC,
  UA=00007700B.
/XYZ1,TT=*ID,RS=6,US=5,DB=BC.
```

Prior to syntax checking by VALNET (described in the following paragraph), this set of directives should be entered on cards or from a terminal and saved as a file.

VALNET VALIDATION

Validate network (VALNET) is a system program that checks the syntax and logic of a terminal network description file.

The format is:

```
VALNET(p1,p2,p3)
```

The p parameters can be in any order and are:

- P Specifies the name under which the terminal network file is stored. Options are:

P	File name is COMPILE.
P=fn	File name fn is supplied by user.
Omitted	Default file name is NETWid where id is the two-character machine identifier. This can be obtained by using the MACHID macro which is described in the NOS Reference Manual, volume 2 (refer to preface for publication number).
- L Specifies the file that receives the list of errors. Options are:

L	File name is LIST.
L=fn	File name fn is supplied by user.
L=0	No list is produced.
Omitted	Default name is OUTPUT.
- NR Specifies whether the network description should be rewound before VALNET activity commences.

NR	No rewind before VALNET reads the network description file.
Omitted	Rewind before reading.

For each error encountered in a network description file, VALNET produces two output lines with the following format.

Error Line	
Statement Number	Diagnostic Message

```

COPYBR(INPUT,NET)
SAVE(NET)
7/8/9
Network description statements
6/7/8/9

```

Output is produced only if errors are encountered. Error messages are described in appendix B.

NETWORK FILE ORIGATION

When an error-free network description is ready, an analyst should create a system file called NCTFid (where id is the two-character machine identifier) from the terminal network description file.

A typical procedure for originating an NCTFid file follows. First, the user establishes a set of network description statements as a local file.

```

NCTF.
USER(usrnam,password,familyname)
CHARGE(chargenumber,projectnumber)

```

Then the user enters the following at the system console.

```

X.DIS.
USER(usrnam,password,familyname)
CHARGE(chargenumber,projectnumber)
GET(NET)
SUI(377777)
DEFINE(NCTFid/CT=PU)
COPY(NET,NCTFid,V)
RETURN(NCTFid,NET)

```

This creates a direct access public file called NCTFid under the system user index 377777. NCTFid is used to generate terminal status table (TST) entries.

Figure 8-1 illustrates a terminal status table entry. Each line gives the user name and its associated data base, read and update securities, and user area value.

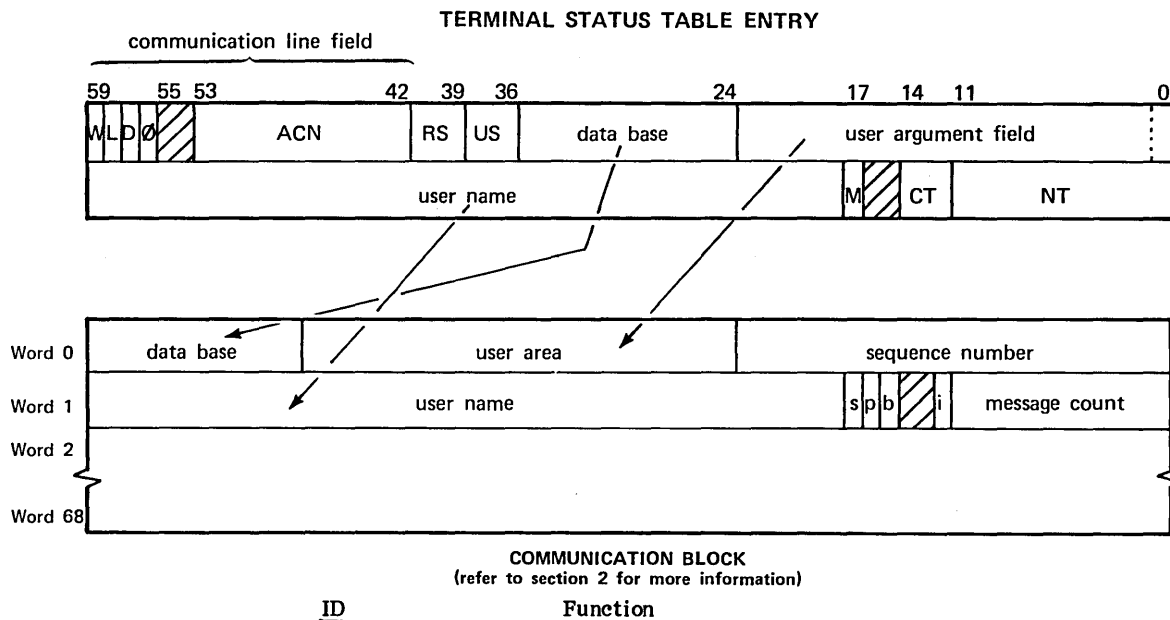


Figure 8-1. Communication Between Terminal Status Table and Communication Block

When a transaction originates from a terminal, the transaction executive uses the appropriate entry in the terminal status table to construct words 0 and 1 of the communication block.

After the network file is originated, it must be updated to reflect changes, and the transaction executive must be reinitialized to recognize the new information.

A task library contains all of the tasks which may be requested by a user group within the transaction subsystem. The task library is created by the LIBTASK utility.

A task library file is a direct-access permanent file with no password that must be created with a DEFINE command before a library creation run occurs. However, it does not have to be attached by the user unless the DA option is specified (refer to LIBTASK control statement). The transaction subsystem simultaneously supports one system task library, and optionally, one task library for each data base.

The system task library, called TASKLIB by default, may be renamed at the user's option. It is defined under the transaction subsystem user number and given write permission to that user number. The name TASKLIB may also be changed using the K display command K.TLF. (Refer to the NOS Operator's Guide.)

A user task library must be named xxTASKL where xx is the name of the data base with which the library is associated. It must be defined under the data base user number, and explicit write permission must be granted to the transaction subsystem user number to access the file.

During initialization, an attempt is made to attach a library for each initialized data base. The xxJ file is accessed to determine the user number needed to attach the library. When a scheduling request is received, the transaction subsystem searches for the task first in the associated data base library, if one exists, and then in the system library. Each user can, therefore, have tasks with the same name, and task security between tasks is still maintained. The only exception to this procedure is the initial task, which is always loaded from the system library.

LIBTASK UTILITY

The LIBTASK utility creates a new library, adds new tasks to an existing library, replaces tasks on an existing library, and compacts an existing library by removing inactive records. Tasks may be added or replaced while the transaction subsystem is running, because tasks are added as a new file at the end of the library. The library is a multifile file. The transaction executive attempts to read a new library directory if the LIBTASK control statement contains the TT option. Thus, on-line update of any task library is allowed.

If the library is modified while the transaction subsystem is running and the user wants the new directory used, the user number and password of the LIBTASK user are passed to the transaction subsystem for validation. This is done by intercontrol point communication, which is also used to check if the transaction subsystem is running if the CR or

PR option is selected. Therefore, the first USER statement in the job must have permission to do intercontrol point communication. (Refer to the NOS System Maintenance Reference Manual for the use of MODVAL to alter permissions.) The LIBTASK run issues an error message and does not update the library if the validation fails. The task binaries to be added or replaced are in ABS or OVL format produced by the CYBER Loader.

TASK RESIDENCY AND TYPE

An important option of LIBTASK enables the user to specify the residency and type of tasks on the task library. A task can reside on central memory, ECS, or mass storage. This can be specified with the C and E options on LIBTASK directives.

A task can be of either reusable or nonreusable type. A reusable task must be written so that it is reinitializing. A reinitializing task (program) does not modify program areas that the program considers constant, or if it does, it resets (reinitializes) these areas upon reentry. Multiple requests for a reusable task can be queued together; this is partly under the control of the person who sets up the task library and partly under the control of the transaction executive. The queuing of multiple requests occurs in an internal list, the Requested Task List (RTL). Each entry in this list represents a scheduling request for a copy of a task. There may be multiple entries in the RTL list for a given task. The LIBTASK queue limit input directive can be used to specify the maximum number of transactions which can be queued against one entry. When the transaction executive schedules a copy of a task from the RTL list, it picks an RTL entry and forces all of the multiple requests queued for that RTL entry to use the same copy of the task.

It is generally possible to have multiple copies of a given task in execution at the same time (note that this does not apply to CM resident tasks). Additional application control of task scheduling is provided by the Q LIBTASK option in the input directives.

A nonreusable task is not a reinitializing task; a new copy of the task must be loaded from the task library for each transaction.

The task type can be specified with the D option on LIBTASK directives. The queue limit can be specified with the QL option.

If a task is specified as central memory resident and reusable, it is loaded to a subcontrol point when TAF is initialized and, unless the task aborts, remains at that subcontrol point to service transactions as long as TAF is running. If the task aborts, another copy of it will be loaded. It is a single copy task. No additional copies of this task are loaded to process requests after queue limit has been reached.

LIBTASK CONTROL STATEMENT

The LIBTASK control statement has two formats. The first format allows specification of the file containing input directive statements; this format is:

LIBTASK(p₁,p₂,...,p_n)

The second format specifies that input directive statements are included in this LIBTASK control statement; this format is:

LIBTASK(p₁,p₂,...,Z,...,p_n)/d₁/d₂/.../d_n.

/ Unique delimiter that does not occur in any d_i input directives. This delimiter does not have to be a slash.

d_i Input directive statement (refer to Input Directives in this section).

The input directives on a LIBTASK statement remain in effect for a task in an existing task library until the task is specified in a subsequent LIBTASK statement; that is, the most recent input directives (including default input directives) for a task take precedence over earlier input directives. However, this is not true for a task that is not in a task library; in this case, the first input directives (including default input directives) take precedence.

For example, the user specifies the following statement:

LIBTASK(p₁,p₂,p₃,...,p_n)/*T1 DL.

This statement logically deletes task T1. If the user specifies task T1 in a subsequent LIBTASK statement and does not specify DL, the task is activated (by default).

The LIBTASK control statement cannot exceed 80 characters.

The p parameters for either format can be in any order and are:

Parameter	Meaning
I	Specifies the file containing input directives. The file is not rewound before processing. All directives that do not refer to a task name present on the B file or P file are ignored. Options are:
I	File is COMPILE.
I=lf _n	File is specified by user.
I=0	No directive file exists.
Omitted	Default file is INPUT.
B	Specifies the file containing task binaries to be added to the library. These may be tasks already existing in the library or they may be new ones. The file is rewound before processing. Options are:
B	File is LGO.
B=lf _n	File is specified by user.
B=0	Normal output information is given.
Omitted	Default file is LGOB.

Parameter	Meaning
L	Specifies the file on which output is written.
L	File is OUTPUT.
L=lf _n	File is specified by user.
L=0	No output is written.
Omitted	Default file is OUTPUT.
P	Specifies the name of a direct-access permanent file to be used as the library file. The file is required. Options are:
P	File is TASKLIB.
P=lf _n	File is specified by user.
P=0	No old library file exists.
Omitted	Default file is TASKLIB.
N	Specifies a newly defined direct-access permanent file to be written as the new library file. Use of this option eliminates the complete copy of the P file to a scratch file on the PR option, thus significantly reducing the purge time for long libraries. The P file is not purged. Options are:
N	File is TASKLIB.
N=lf _n	File is specified by user.
Omitted	Default file is the file specified by P.
DA	Disables all attaches and returns of the library files in the LIBTASK program. With this option selected, the N file and the file specified by P must be attached prior to the LIBTASK call. If a library is attached in the proper mode from another user name, the DA option must be selected so that LIBTASK does not try to attach the appropriate libraries under the user name from which the update is being run.
CR	Creates a new task library. LIBTASK reads one task at a time from the B file, checks for an input directive for the task, constructs a directory entry, and inserts the entry in alphabetical order in the library directory. The directory is written at the end of the library. This option applies only if LIBTASK was able to attach the task library (specified by the N option) in write mode.
PR	Removes inactive task binaries and old directories from the task library. LIBTASK processes the B file and the input directives normally, and if N is not specified, copies the entire library to a

<u>Parameter</u>	<u>Meaning</u>
	scratch file. The current binary for each task in the library is then transferred from the scratch file to the library file, and the directory is updated and written at the end. Therefore, only one copy of each task and one directory remain in the library. If N is specified, no scratch file is written, and P is not purged. The new library file contains only the latest copy of each active task and one directory.
	This option applies only if LIBTASK was able to attach the task library (specified by the N option) in write mode.
TT	Instructs the transaction executive to update the TAF resident copy of the task library directory while TAF is running. If not specified, TAF will not become aware of the update until the next time it is initialized.
	The installation parameter TLDL is used to specify the amount of space, in words, to be reserved for additions to each TAF resident task library directory. The default value is 30 which allows 10 additions (3 words for each TLD entry). Although further additions can be made to the user's task library, they will not be recognized by TAF until TAF is reinitialized. There is no limit to the number of times existing tasks can be replaced.
	The user number from which LIBTASK is being run must be the same as the user number in the xxJ file associated with the library being modified, and TAF must be able to attach the library in WRITE mode in order for the update of TAF's copy of the TLD to take place. The user must be validated for intercontrol point communication (the CSTEP bit in user's access word must be set) to use the TT option.
Z	Specifies that input directives are on the LIBTASK statement. This option overrides the I option.

INPUT DIRECTIVES

To alter task characteristics, directive statements may be entered in the directive file. Only one statement should appear for each task. These characteristics are kept in the directory for use by the transaction executive when the task is to be executed.

The directive statement format is:

*name p₁,p₂,...,p_n.

name is the task name and p_i is any of the following input directives in any order.

<u>Directive</u>	<u>Meaning</u>
BP=nn	Beginning scheduling priority of task.
BP	Beginning priority defaults to 25B.

<u>Directive</u>	<u>Meaning</u>
BP omitted	Beginning priority defaults to 20B.
C	Specifies a memory-resident task. The default is a disk-resident task.
D	Specifies a destructive code (non-reusable) task that is removed from the subcontrol point when the task is completed. The default is a reusable task. This option is recommended when debugging a task.
DL	Specifies that a task (name) is logically deleted. All logically deleted tasks are physically deleted from the task library when the LIBTASK statement contains the PR option.
E	Specifies an ECS-resident library copy of the task. The default is mass storage resident.
	If tasks are specified as ECS-resident, the transaction executive during initialization loads as many as will fit into its available ECS and leaves the remaining tasks on the mass storage file. If no ECS is available, all tasks are loaded from mass storage.
	If a LIBTASK function specifying the TT option is issued while the transaction executive is running, no tasks are replaced or added to ECS, and one of the following situations occurs.
	• Old copy of task in ECS and new copy in ECS. The new copy is ignored, and the old copy remains on the library in ECS. • Old copy not in ECS and new copy in ECS. The new copy is used but is read from mass storage. • Old copy in ECS and new copy not in ECS. The new copy is loaded from mass storage.
EF=efl	Specifies the maximum amount by which a task will be allowed to increase its initial field length. efl is assumed to be octal unless it contains an 8 or 9 or is followed by a trailing D. The default value is zero. EF will be increased to the next multiple of 100g if it is not already such a multiple. The maximum field length (MFL) allowed for the task is (initial task FL) + efl. The value of efl can range from 0 to 300000g.
IG	Specifies that a task (name) in the file specified with the B option is ignored.
MP=nn	Maximum scheduling priority of task (for future use in aging tasks).
MP	Maximum priority defaults to 50B.
MP omitted	Maximum priority defaults to 40B.

<u>Directive</u>	<u>Meaning</u>
O	Sets task to OFF; requests for the task are routed to the OFFTASK routine (refer to OFFTASK in section 10). The default is ON.

Q	Specifies that queuing is to be forced as described below. This option alters the normal scheduling priority of the task (refer to Scheduling Priority) as follows:
---	---

1. When there are no active copies of the task, the first copy of the task is scheduled as usual (refer to Scheduling Priority).
2. While there are from one to six active copies of the task in TAF, another copy of the task is not scheduled until the number of transactions queued against its RTL entries equals or exceeds the queue limit specified by the QL directive. When an RTL entry is used to schedule a task, it is released and made available for requesting the scheduling of other tasks.
3. When the number of active copies of the task equals 7, additional copies of the task will not be scheduled. This applies even if the queue limit is 1.

An active task is a copy of a task which is both:

- At a subcontrol point or rolled out
- In the process of handling a transaction or has transactions queued against it (the copy).

If a LIBTASK function specifying the TT control statement option is issued while the transaction executive is running and an attempt is made to change the Q-attribute of a task on a library in use by TAF, then the Q-attribute change will not affect TAF scheduling until the executive is reinitialized.

This parameter is not applicable to CM resident tasks.

QL=n	Specifies the number of requests, up to a maximum of five, that can be queued for an RTL entry for a task. If a task uses a large amount of memory and executes rapidly, performance is not degraded if requests are queued. In this case, multiple copies of the task require too much memory. If a task does a large amount of I/O, then performance is degraded if requests are queued. In this case, multiple copies of the task enhance performance.
------	---

QL	Queue entry limit defaults to five.
----	-------------------------------------

<u>Directive</u>	<u>Meaning</u>
QL omitted	Queue entry limit defaults to three.
R	Specifies that the FL assigned to a CM resident task is to be reduced to the initial load value each time the transaction is run. The default is no reduce. Non-CM resident tasks are always started with their initial load value.

SC	Sets the task communication block load status to solicited. This means the user must request the communication block and supply a valid address (using the BEGIN request) after the task is initiated. The default is unsolicited.
----	--

All directives for one task must appear on the same line. The following situation will result in only p₁ being recognized as an input directive for TASK1.

*TASK1 p ₁ .	
*TASK1 p ₂ .	
.	
.	
*TASK1 p _n .	

Improper way to specify input directives.

SCHEDULING PRIORITY

When the tasks are queued, the scheduler will attempt to load the task with the highest current priority. If there are several tasks with the highest priority, the task with the lowest field length requirement will be selected for loading. When a task enters the queue, the beginning priority (BP) is used as the tasks current priority. The current priority is then aged (incremented by one) every second up to the maximum priority (MP). Tasks in the queue having identical current priorities and field length requirements normally will be scheduled in the order they entered the queue.

With these directives, classes of tasks can be created. For example, if two classes were created (X and Y) their priorities could be:

X _{BP} = 25 _g	X _{MP} = 40 _g
Y _{BP} = 30 _g	Y _{MP} = 45 _g

In this case, no tasks in X could be scheduled until all tasks in Y having a current priority greater than 40_g had been scheduled.

CREATING A NEW TASK LIBRARY

The following examples show how a user can create a new task library. An explanation is provided after each LIBTASK statement.

LIBTASK(P=0,N=lf_{n1},CR,B=lf_{n2},I)

LIBTASK copies file lf_{n2} to file lf_{n1} according to input directives in the COMPILE file and adds a new sorted task library directory at the end of file lf_{n1}.

LIBTASK(P=lf_n2,N=lf_n1,Z)/*name p₁,p₂,...,p_n.

LIBTASK copies file lf_n2 to file lf_n1 and adds a new sorted task library directory at the end of file lf_n1 according to input directives on the LIBTASK statement.

LIBTASK(P=lf_n2,N=lf_n1,PR,Z)/*name p₁,p₂,...,p_n.

LIBTASK copies all active (not logically deleted or replaced) task binaries from file lf_n2 to file lf_n1 and adds a new sorted task library directory at the end of file lf_n1 according to input directives on the LIBTASK statement.

LIBTASK(P=lf_n2,N=lf_n1,B=lf_n3,I)

LIBTASK copies file lf_n2 to file lf_n1, copies file lf_n3 to file lf_n1 according to input directives in the COMPILE file, and adds a new sorted task library directory at the end of file lf_n1.

LIBTASK(P=lf_n2,N=lf_n1,B=lf_n3,PR,I)

LIBTASK copies all active (not logically deleted or replaced) task binaries from file lf_n2 to file lf_n1, copies file lf_n3 to file lf_n1 according to input directives in the COMPILE file, and adds a new sorted task library directory at the end of file lf_n1.

UPDATING AN EXISTING TASK LIBRARY

The following examples show how a user can update (add new tasks and delete and replace existing tasks) an existing task library without supplying any binaries. An explanation is provided after each LIBTASK statement.

LIBTASK(P=lf_n1,Z)/*name p₁,p₂,...,p_n.

LIBTASK adds a new sorted task library directory at the end of file lf_n1 according to input directives on the LIBTASK statement.

LIBTASK(P=lf_n1,PR,Z)/*name p₁,p₂,...,p_n.

LIBTASK copies all active (not logically deleted or replaced) tasks from file lf_n1 onto a scratch file, copies the scratch file back onto file lf_n1, and adds a new sorted task library directory to the end of file lf_n1 according to input directive on the LIBTASK statement. This statement is not valid when TAF is up and has attached file lf_n1.

LIBTASK(P=lf_n1,B=lf_n2,Z)/*name p₁,p₂,...,p_n.

LIBTASK copies file lf_n2 to the end of file lf_n1 according to input directives on the LIBTASK statement. LIBTASK adds a new sorted task library directory to the end of file lf_n1.

LIBTASK(P=lf_n1,B=lf_n2,PR,I=lf_n3)

LIBTASK copies all active (not logically deleted or replaced) tasks from file lf_n1 onto a scratch file, copies the scratch file back onto file lf_n1, copies file lf_n2 to the end of file lf_n1 according to input directives on file lf_n3, and adds a new sorted task library directory to the end of file lf_n1. This statement is not valid when TAF is up and has attached file lf_n1.

DELETING A TASK FROM TASK LIBRARY

With the DL input directive on the LIBTASK statement, the user can logically delete a task from the task library if TAF is up and has attached the task library as follows:

LIBTASK(P=lf_n1,TT,Z)/*name DL.

In this example, LIBTASK logically deletes a task (name) from file lf_n1. The TT option is required when TAF is up. The DL input directive is the only input directive allowed for the named task.

With the PR option on the LIBTASK statement, the user can delete inactive tasks from the task library when TAF is not up and when TAF is up but has not attached the task library.

LIBTASK(P=lf_n1,PR,Z)/*name DL.

In this example, LIBTASK physically deletes a task (name) from file lf_n1. TAF must not have the file attached when this statement is executed.

IGNORING A TASK IN THE REPLACEMENT FILE

With the IG input directive on the LIBTASK statement, the user can tell LIBTASK to ignore a task binary in the replacement file specified with the B option as follows:

LIBTASK(P=lf_n1,B=lf_n2,Z)/*name IG.

In this example, LIBTASK skips a binary task (name) on file lf_n2 when copying file lf_n2 to file lf_n1. The IG input directive is the only input directive allowed for the named task.

The applications programmer should be aware of the functions of these tasks but may omit the detailed information contained in this and section 11.

Several system tasks are required with every application: initial task (ITASK), K-display director (KDIS), logout transaction terminal (LOGT), message abort (MSABT), inactive task (OFFTASK), system message (SYSMSG), and execute named tasks (XTASK). Each must reside on the system task library of every transaction system. Limited versions of these system tasks are supplied by Control Data; however, modifications to these tasks are often necessary for specific applications.

Directions for installing these tasks are contained in the TAF Installation section of the NOS Installation Handbook.

INITIAL TASK (ITASK)

The primary functions of ITASK are analyzing terminal input and TAF-originated transactions. If the TAF-originated transaction field is set in the communication block (figure 2-1), ITASK uses a table to initiate the processing routine associated with that type of transaction. Otherwise, ITASK examines three characters, starting with the first character of the input message. The first of these three characters is the transaction code; the other two comprise the subtransaction code entered from a terminal.

Tables in ITASK associate combinations of transaction and subtransaction codes with task names. When a valid combination is received, the associated task name is requested for scheduling via a CALLTSK request. If an illegal combination occurs, MSABT is requested to send an error message to the terminal. It is not possible to have a transaction code without a subtransaction code entered from a terminal. However, spaces can be defined as valid subtransaction codes.

ITASK recognizes the transaction code EX., which is used to call the system task XTASK. This special transaction code is reserved for XTASK.

NOTE

Because ITASK schedules tasks, care must be taken to avoid entering an infinite loop when a task calls ITASK.

ITASK is described in more detail in section 11.

KDIS

KDIS is a system task scheduled by the transaction executive when the K.SWITCH command is entered by the console operator. KDIS requests the K display and displays a complete list of K display commands. The K display commands are discussed in the NOS Operator's Guide.

LOG-OUT TASK (LOGT)

This task logs a terminal out of TAF. However, the task does not log the terminal off the computer system. The task sends a message, END OF TRANSACTION SESSION, performs a LOGT request, and performs a CEASE request. The terminal is then available for login to TAF or other network applications.

MESSAGE ABORT (MSABT)

Message abort (MSABT) is a system task that sends error messages to the originating terminal when a transaction is terminated due to a system-detected error. MSABT checks for error codes that were placed in the communication block by the transaction executive or the initial task. When an error code is detected, MSABT sends the appropriate error message and initiates an abnormal CEASE function.

MSABT is scheduled in the same manner as other tasks.

If the Total data manager is used, MSABT must be modified to perform a FREEX command. This ensures that Total records are released if a task aborts.

Error messages issued by MSABT are described in appendix B.

OFFTASK

This task is provided to give the user control over an internal condition which might be the result of operational error. OFFTASK is scheduled by the transaction executive when a request has been received for an inactive task. The task may be inactive or off due to the use of the K.OFFTASK command or use of a LIBTASK directive when the task was placed in the library. OFFTASK sends an error message to the originating terminal and ceases abnormally. It is expected that the user will expand the task to handle any specific conditions that may exist when a transaction is interrupted by an inactive task.

SYSMMSG

This task sends K.MESSAGE-entered messages and K display diagnostic messages to terminals. It calls the subroutine in COMKCBT passing the user argument bits. The word returned is sent after the message is sent.

EXECUTE NAMED TASK (XTASK)

Execute named task (XTASK) is a special task scheduler used to execute most named tasks. It will not execute tasks by the names ITASK, KDIS, MSABT, OFFTASK, SYSMMSG, and XTASK. These are the names of required system tasks. XTASK can be used to schedule LOGT as a means of logging a terminal out of transaction mode. XTASK can be called via ITASK by the following input:

EX.task

EX.	Special three-character transaction code which causes ITASK to schedule XTASK.
task	One- to seven-character alphanumeric task name, left-justified (no space between EX. and task) with a nonalphanumeric character terminator.

A site may want to delete the TRANS table entry of STRAN (X.), XTASK in the TRANT table of ITASK, or the system task XTASK altogether for security reasons. XTASK was provided as a means of scheduling tasks, and as an example of how a site might use such a task. However, it may provide too much flexibility, and therefore should be used with discretion.

This section describes some of the features, options, and requirements of ITASK.

MACROS

ITASK contains three tables. These tables are set up for the following codes.

- Time a task is to be run
- Transaction code
- Subtransaction code

A macro is used to set up each type of ITASK table. These three COMPASS macros, TIMCNT, TRAN, and STRAN, are discussed here.

TIMCNT MACRO

The TIMCNT macro makes an entry in the table of time-originating tasks (TTOT). TTOT contains the time a task is to be initiated and an address.

One TTOT contains all of the entries, regardless of the application that is to use them. This feature allows tasks to be run at predetermined times without specific task requests or terminal input to initiate the run.

The format is:

```
TIMCNT hh,mm,ss,addr
      hh      Hour
      mm      Minute
      ss      Second
      addr    Routine address
```

The TTOT table format is:

```
Word 1      TTOT, number of entries in the table
Word 2      Table entries
and following
```

The format of an entry is shown in figure 11-1.

TRAN MACRO

The TRAN macro sets up entries for the table of transaction codes (TRANT). TRANT contains the address of the table of subtransaction codes for that particular transaction code. ITASK assumes that the TRANT table is set up in display code order. It uses the display code value of the transaction code entered by the user to locate the proper entry in TRANT.

The format is:

TRAN code

code. One-character transaction code

The format of an entry in TRANT is shown in figure 11-2.

STRAN MACRO

The STRAN macro makes entries in the table of subtransaction codes (TRANS). TRANS contains the subtransaction code and the corresponding task name.

The format is:

STRAN sc, task

sc Two-character subtransaction code

task Name of the task to be called

The format of an entry in TRANS is shown in figure 11-3.

Each group of entries relating to the same transaction is placed immediately following the TRANT entry for that code, although the subtransaction table is assembled elsewhere. This is an example of the TRANT and TRANS tables.

```
TRAN BSS 0
TRAN A
STRAN 02,TSKNAME
STRAN 03,TASKN
```

A is the transaction code, 02 and 03 are the subtransaction codes, and TSKNAME and TASKN are the respective tasks to be requested.

A site may want to delete the TRANS table entry of STRAN (X.), XTASK in ITASK for security reasons. XTASK is described in more detail in section 10.

ITASK DESCRIPTION

Entries in all ITASK tables, as well as the user subroutines for TIMCNT entries in TTOT, should be inserted in the designated places within the program before assembly. TRANT and TRANS entries must be in alphabetical or numerical order according to display codes.

If ITASK receives a valid transaction code and subtransaction code, the call is made to the desired task. If the codes are invalid, the terminal receives an appropriate message from MSABT.

TTOT entries must be accompanied by user-written subroutines to set up and call the task, as follows:



addr Address is the tag of a user-written subroutine that is called by ITASK to set up and call the desired task.

ITASK then clears the table entry so the request is not repeated.

hhmmss Time the task is to be initiated, in hours, minutes, and seconds.

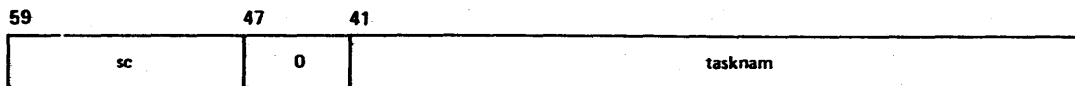
Figure 11-1. Format of TTOT Table Entry



length Length of subtransaction code table (in format allowing unpack function to retrieve it).

address Address of section in subtransaction code table containing associated entries.

Figure 11-2. Format of TRANT Table Entry



sc Subtransaction code.

tasknam Task name, left-justified, zero-filled.

Figure 11-3. Format of TRANS Table Entry

- The system time is retrieved from the appropriate word in the communication block. TTOT is then searched for any entries that were to be run by that time.
- If such an entry is found, ITASK jumps to the user's subroutine to set up and call the task. This subroutine must set up the communication block required by the task that is to be run. It must perform a CALLTSK without the cease option set to ensure that control is returned to ITASK.
- Transaction executive recovery.
- Terminal login.
- Network and terminal failures.

When any of these conditions occurs, a TAF-originated transaction may be processed. Code in common deck COMKSTC identifies the condition. ITASK examines the communication block (figure 11-4) to determine the condition and jumps to a subroutine which sets up and calls the proper task or sequence of tasks to respond properly to the identified condition. Although the Control Data-supplied ITASK contains code to process each of the conditions, sites may want to modify ITASK and provide transactions to suit their individual needs.

TAF-ORIGINATED TRANSACTIONS

TAF recognizes certain conditions not normally noted or processed by user applications. Some of these conditions are:

- Exceeding a length of time as specified by an assembly option (TACTL).
- K.IDLE command telling the transaction subsystem to idle down.

To provide TAF-originated tasks that properly respond to the NAM conditions, the analyst should refer to the NAM Reference Manual.

A number of common decks aid in processing the TAF-originated transactions as follows:

- COMKSTC gives the values for the condition field of the communication block.
- COMKNWF describes the fields for NAM messages word of the communication block.
- COMSNCD gives values to fields in NAM messages word.

CONDITIONS REQUIRING TAF-ORIGINATED TRANSACTIONS

The conditions that require TAF-originated transactions and possible methods of processing are described here. Limited versions of routines needed to process these transactions are available in ITASK. To change the existing routines, ITASK must be modified to include the necessary code or call the necessary tasks. It may be advisable to include short procedures as subroutines in ITASK, but long or frequently updated transactions should be maintained as separate programs apart from ITASK and merely called from the ITASK subroutine.

Time Activation

This might be processed by ITASK as follows:

1. ITASK retrieves the system time from the appropriate word in the communication block.
2. ITASK then searches the TTOT for any transactions that are to be run at that time.
3. If such an entry is found, ITASK jumps to the user-supplied subroutine that must set up and call the next task for that transaction. This subroutine must set up the communication block required by the task that is to be run.

K.IDLE Console Operator Command

The analyst may want to provide tasks that prevent new users from logging in or provide transactions from the beginning.

TAF Subsystem Recovery

TAF recovery is initiated when sense switch 4 is set. ITASK is scheduled before communications with terminals begin. Recovery tasks may be initiated at this point.

Terminal Login

The application may want to prompt the terminal user, set application character type, or go through recovery procedures if the user has the recovery field set in the

terminal status table. ITASK should not perform any time-consuming processing itself since this may cause transactions to be queued for ITASK. Rather, ITASK should call another task to perform this time-consuming processing, freeing ITASK for other transactions. Upon terminal login, the NAM message word of the communication block (figure 11-4) contains the following information:

<u>Bits</u>	<u>Significance</u>
35-25	Block size of terminal
24	Signifies a hardwired terminal if set
23-16	Terminal class
15-8	Page width of terminal
7-0	Page length of terminal

Terminal Break Conditions

If some tasks perform SENDs without the recall flag set, breaks can occur because of the following.

- User break keys
- Output device not ready
- Incorrectly formatted data

For a user to resume processing after a terminal break, the following must occur.

1. Terminal communication with NAM must be reestablished. When the break occurs, NAM places the supervisory message associated with the break in the NAM message word of the communication block. To restart terminal communications, TAF must send a reset supervisory message to NAM. The individual application need not provide this function. Upon the user task SEND request, TAF issues the reset supervisory message prior to honoring the SEND.
2. The user must resume interaction with the application. The user application must provide this break recovery. The following description suggests one way.

ITASK and any recovery tasks can communicate through the terminal status table user area. For instance, when ITASK initiates a transaction, it could set a series of fields in the TST to indicate important events in the transaction. The recovery task could examine the fields to determine what events had occurred. Nonrecovery tasks could watch a recovery field in the TST and CEASE to allow the recovery task to proceed in a quiet environment. (TAF sets the recovery field in the TST before calling ITASK.)

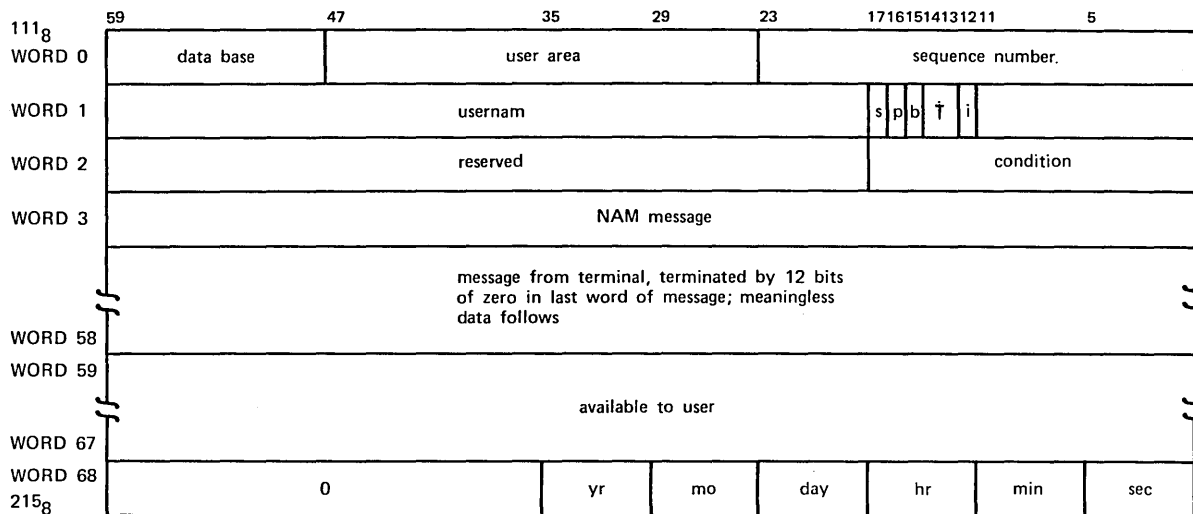


Figure 11-4. Communication Block for TAF-Originated Transactions

Terminal Connection Broken

A connection may be broken because a line has been disconnected. If a SEND with recall was active, the task issuing the SEND rather than ITASK is notified of the error. It may be desirable for the application to process these incomplete transactions. TAF sets the recovery field in the terminal status table user area. The application can save terminal output on a data manager file and send it when the user logs in again. The outstanding block limit specified in the local configuration file determines how many messages must be saved. (Refer to the NDL Reference Manual.)

Network Normal Shutdown

TAF monitors task activity on communication blocks generated by terminal input. When all tasks have been completed and no tasks are performing WAITNP requests, TAF sends end connection supervisory messages to NAM for each logged-in user and performs a NETOFF to end processing with the network. To initiate this condition, it may be desirable for ITASK to notify users of the pending shutdown and begin rejecting new transactions. A console operator K.NAMON command informs TAF to perform a NETON to initiate communications with the network.

Network Abort

TAF sets the recovery field in the terminal status table for all logged-in users. All tasks performing WAITNP requests are scheduled. The application, through ITASK, may want to begin data base recovery.

Network Immediate Shutdown

TAF performs a NETOFF and sets the recovery field in the terminal status table. The application may want to begin recovery by processing only transactions from batch input.

Terminal Inactive

NAM issues a terminal inactive supervisory message when it detects that 10 minutes have elapsed without message traffic over a connection. TAF passes the transaction number of any running task using the user name in the sn field of the communication block. The application may want to log out the user after a period of time to reduce line connect costs.

Logical Error

One of the application's tasks may have set up an application block header or message block incorrectly. If the task performed a SEND request with recall set, the task, rather than ITASK, is notified of the error.

Block Not Delivered

This occurs because of a system error. If the task performed a SEND request with recall set, the task, rather than ITASK, is notified of the error. Before scheduling ITASK, TAF sets the recovery field in the TST. It may be desirable for the application to recover by sending the lost block. The application block number of the block that was lost is returned in the NAM message word of the communication block. Tasks may specify a block number on the SEND or obtain the block number used by TAF. Resending the lost block may cause the block to be delivered out of sequence, if further blocks had been sent following the lost block.

Terminal Characteristic Changes

A NAM terminal characteristic supervisory message is sent to ITASK when the terminal operator has redefined the terminal class, page width, or page length.

Terminal Input Too Large

The terminal operator sent a message longer than the maximum input message size. TAF will not have journalized the input.

Stop and Start on Downline Connection

A STOP supervisory message from NAM occurs because the terminal is busy or is failing. The network does not process downline blocks for the user until the network issues a start on downline connection supervisory message. The following approach may be used on a downline stop condition.

1. NAM sends the stop on downline connection supervisory message to TAF.
2. TAF sets the stop field in the network communication table (NCT) so that TAF may either abort tasks currently performing SEND requests or return a status to the user.
3. TAF sets the recovery flag for the user in the TST.
4. TAF then schedules a TAF-originated transaction to explain the reason for stoppage.
5. ITASK then begins to perform TSIM requests to check the recovery field in the TST. If the recovery field is set, ITASK either rejects or saves the input from the user.
6. TAF receives a start supervisory message from NAM.
7. TAF clears the stop field in the NCT to allow SEND requests. TAF then schedules ITASK with a start on downline connection reason code. ITASK then sends a message explaining the reason for the interruption. The next SEND to the user causes TAF to send a RESET supervisory message to NAM before the SEND is performed.

8. ITASK then schedules a recovery task to send an error message to the user and performs a WAITINP request to resequence the transaction.
9. The recovery task then corrects the data base.
10. The recovery task then clears the recovery field in the TST.
11. ITASK then accepts input from the user.

TERMINAL TYPE AHEAD

The network allows terminal operator type-ahead. In other words, the terminal operator can enter several transactions without waiting for the response from the first.

USER STATUS

The network allows a terminal user to make a status request by typing:

(CT) (ALPHA) ⒸⒶ

(CT) The control character

(ALPHA) Any alphabetic character

TAF will right justify the character in word 3 of the communication block and initiate ITASK. At this point a user-written routine in ITASK must take over to perform whatever action is desired. The ITASK reason code will indicate a user status message.

CHARACTER SETS

A

Table A-1 lists the character codes for the 63- and 64-character sets. Users should check with a site analyst to see which character set is being used.

TABLE A-1. ASCII CODE STANDARD PRINT 63- AND 64-CHARACTER SET

ASCII CHAR.	ASCII CODE		DISPLAY CODE (OCTAL)
	OCTAL	HEXADECIMAL	
:	072	3A	00†
A	101	41	01
B	102	42	02
C	103	43	03
D	104	44	04
E	105	45	05
F	106	46	06
G	107	47	07
H	110	48	10
I	111	49	11
J	112	4A	12
K	113	4B	13
L	114	4C	14
M	115	4D	15
N	116	4E	16
O	117	4F	17
P	120	50	20
Q	121	51	21
R	122	52	22
S	123	53	23
T	124	54	24
U	125	55	25
V	126	56	26
W	127	57	27
X	130	58	30
Y	131	59	31
Z	132	5A	32
0	060	30	33
1	061	31	34
2	062	32	35
3	063	33	36
4	064	34	37
5	065	35	40
6	066	36	41
7	067	37	42
8	070	38	43
9	071	39	44
+	053	2B	45
-	055	2D	46
*	052	2A	47
/	057	2F	50
(	050	28	51
)	051	29	52
\$	044	24	53
=	075	3D	54

† IN THE 63-CHARACTER SET, THIS DISPLAY CODE REPRESENTS A NULL CHARACTER. ALSO, USE OF THE COLON IN PROGRAM AND DATA FILES MAY CAUSE PROBLEMS. THIS IS PARTICULARLY TRUE WHEN IT IS USED IN PRINT AND FORMAT STATEMENTS.

3AE15A

TABLE A-1. ASCII CODE STANDARD PRINT 63- AND 64-CHARACTER SET (Contd)

ASCII CHAR	ASCII CODE		DISPLAY CODE (OCTAL)
	OCTAL	HEXADECIMAL	
(SPACE)	040	20	55
,	054	2C	56
.	056	2E	57
#	043	23	60
[	133	5B	61
]	135	5D	62
%	045	25	63†
"	042	22	64
—	137††	5F	65
!	041	21	66
&	046	26	67
'	047	27	70
?	077	3F	71
<	074	3C	72
>	076	3E	73
@	100	40	74
\	134	5C	75
^	136	5E	76
;	073	3B	77

† IN THE 63-CHARACTER SET, THIS DISPLAY CODE REPRESENTS A COLON (:), 7-BIT ASCII CODE 072, 7-BIT HEXADECIMAL CODE 3A.

†† ON TTY MODELS HAVING NO UNDERLINE, THE BACKARROW (←) TAKE ITS PLACE.

3AE16A

ERROR MESSAGES

B

This appendix contains an alphabetical listing of messages which may be important to the TAF programmer. Each message is followed by an explanation of the message and/or the circumstances causing it to be issued, the recommended action, and the routine which issued the message.

Lowercase letters are used within a message to identify variable fields. All messages beginning with lowercase (variable) fields are listed alphabetically according to the first nonvariable field after the messages that begin with nonvariable fields.

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
AAM FILES CLOSED.	TAF CRM has closed all AAM files.	None.	TAF
ABORT OF CDCS DETECTED.	Self-explanatory.	None.	TAF
AIP DEBUG OPTION TURNED OFF.	Self-explanatory.	None.	TAF
AIP DEBUG OPTION TURNED ON.	Informative message. All network messages will be placed on trace file ZZZZZDN.	None.	TAF
AIP TOO LARGE FOR LOADING.	A fatal error occurred causing TAF to abort.	Inform site analyst. TWFA must be increased in deck COMKTAF.	TAF
APPLICATION ERROR - NO TERMINAL OUTPUT.	A task error. A transaction has run without either an error message or a SEND being sent to the originating terminal.	Correct task or inform site analyst.	MSABT
APPLICATION FAILED.	The application has ceased functioning and the user can no longer access it.	Access another application or log off and try later.	NETVAL
APPLICATION NOT PRESENT.	The requested application is currently not available for use.	Enter the name of another application or log off and try later.	NETVAL
APPLICATION RETRY LIMIT.	Four unsuccessful attempts were made to enter a legal application name, after which the terminal was disconnected.	Ensure the accuracy of the entry. If problems persist, contact installation personnel.	NETVAL
ARITHMETIC OR OTHER SOFTWARE ERROR IN TASK.	A task error. Hardware or the system detected a software error in the task.	Correct task or inform site analyst.	MSABT
ATTACH ERROR ON - filename.	The transaction executive cannot attach the file filename under present conditions. This usually implies that the file does not exist or permission has not been given to the TAF user name.	Correct error and reinitialize executive, or contact site analyst.	TAF
ATTACH MODE MUST BE W, M, R, OR RM.	The mode parameter on the CRM statement must be one of the specified values.	Correct the mode parameter on the CRM statement or inform site analyst.	TAF
BINARY RECORD WAS LESS THAN 64D WORDS LONG.	The task binary file contains records in the wrong format.	Ensure that a LOAD was performed after compilation of the tasks.	LIBTASK
BINARY RECORD WAS NEITHER ABSOLUTE NOR OVERLAY FORMAT.	The task binary file contains records in the wrong format.	Ensure that a LOAD was performed after compilation of the tasks.	LIBTASK

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
BLOCKAGE AMONG CM RESIDENT TASKS.	The sum of initial field lengths for the CM resident tasks exceeds the minimum size of total task area.	Correct error.	TAF
CALLRTN NESTED CALL LIMIT EXCEEDED.	A task error. An attempt was made to nest CALLRTN requests more than 16 levels.	Correct task or inform site analyst.	MSABT
CDCS NOT AVAILABLE.	Your transaction attempted to use CDCS when it was not present in the system, CDCS aborted while your transaction was waiting for completion of a CDCS request (SSC), or CDCS aborted while your transaction was rolled out.	Inform data base administrator.	MSABT
CDCS REQUESTED ABORT.	TAF aborted the transaction as a result of a request by CDCS. The detailed error message returned by CDCS is available in the JOUR0 file.	Inform data base administrator.	MSABT
CHANGED TLD DETECTED - filename, username.	An unrecognizable library directory format was encountered during a library directory update attempt.	Inform site analyst.	TAF
CONFLICTING LIBTASK OPTIONS - p1,p2.	The user specified an illegal combination of LIBTASK control statement options. Possible illegal combinations of p1 and p2 are: CR,PR CR,TT PR,TT TT,N Another possible illegal combination gives P=0,-N for p1,p2. This means that the user specified P=0 but did not specify N.	The user must choose either p1 or p2. For the P=0,-N combination, the user must specify N if no old task library (P=0) exists.	LIBTASK
CONNECTION PROHIBITED, TRY LATER.	The network is configured so that only one user with a given user name and family name can access the requested application at one time.	Request a different application or log off and try later.	NETVAL
CONNECTION REJECTED.	The requested application is already servicing the maximum number of terminals.	Request a different application or log off and try later.	NETVAL
CONTROL CARD ARGUMENT ERRORS.	Errors exist in the parameters on the LIBTASK control statement.	Correct and rerun.	LIBTASK
CRM(...parameter-list...)	This is a copy of a CRM statement that is in error. A subsequent message follows.	Inform site analyst.	TAF
CRM DATA MANAGER SUCCESSFULLY LOADED.	Self-explanatory.	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
CSM - ILLEGAL COMMUNICATION FUNCTION.	An illegal or unrecognizable request was received by the transaction executive from the CPU monitor.	Inform site analyst.	TAF
DATA MANAGER NOT LOADED.	The TAF data manager was called but has not been loaded.	Inform site analyst.	MSABT
DIRECTORY IS FULL-NO MORE ADDITIONS ALLOWED.	The maximum length of the task library directory has been reached.	LIBTASK must be reassembled to allow for a larger directory.	LIBTASK
DISPLAY DUMP NOT ALLOWED TO TERMINAL.	A display code dump to a terminal is not allowed.	Specify the 0 option on the KTSDMP control statement if a dump to a terminal is desired.	KTSDMP
DSP ERROR ec RETURNED ON JOB ROUTE.	DSP has informed the transaction subsystem that an error was detected in routing a job for task dump processing. ec is the DSP error code.	Inform site analyst.	TAF
DSP ERROR ec RETURNED ON TASK SUBMIT.	DSP has informed the transaction subsystem that an error other than a job or user statement error was detected. ec is the DSP error code.	Inform site analyst.	TAF
DUAL AND TRACE FLAGS FOR FILE filename.	It is illegal to dual-record and trace the same file.	Correct error and reinitialize executive (TAF) or rerun job (BDMI).	TAF BDMI
DUPLICATE DATA BASE IN TCF - xx.	Active data base identifier, xx, in the TCF is not unique.	Fix TCF so that xx appears only once among active (ON) DMS statements.	TAF
ECS READ ERROR.	Self-explanatory.	Inform customer engineer.	TAF
ECS TASK tasknam NOW MS RESIDENT.	Task tasknam could not be loaded into ECS because of insufficient storage. It is loaded into mass storage.	If task must be resident in ECS, more ECS space must be allocated for the TAF user name. Refer to the NOS Installation Handbook.	TAF
ECS WRITE PARITY ERROR ENCOUNTERED.	Self-explanatory.	Inform customer engineer.	TAF
EDT CARD NOT USED FOR CRM.	The xxJ file cannot contain an EDT statement for TAF CRM.	Remove the EDT statement from the xxJ file or inform site analyst.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
ERROR IN LOADING AAMI.	The loader encountered errors while loading the TAF CRM AAM interface (AAMI).	The site analyst should consult the CYBER Loader Reference Manual (listed in the preface).	TAF
ERROR IN LOADING HASH CODE filename.	The loader encountered errors while loading the hashing routine code that is on file filename.	The site analyst should consult the CYBER Loader Reference Manual (listed in the preface).	TAF
ERROR IN LOADING NAM.	The library file NETIO does not contain the entry points required for TAF.	The site analyst should consult the CYBER Loader Reference Manual (listed in the preface).	TAF
ERROR IN LOADING TOTAL.	The loader encountered errors while loading Total and the data base descriptor modules (DBMODs).	The site analyst should consult the CYBER Loader Reference Manual (listed in the preface).	TAF
ERROR IN READING TASKLIB-filename.	Error occurred during transaction executive initialization or ECS-resident task loading. File specified as task library was incorrectly formatted; therefore, it could not be read or loaded into ECS correctly.	Inform site analyst.	TAF
ERROR IN SUBMITTED FILE.	An illegal parameter has been detected on a SUBMT call from a task.	Correct task or inform site analyst.	MSABT
ERROR ON xxJ FILE ARGUMENTS.	The xxJ file contains statements in error, which causes the transaction subsystem to abort.	Examine xxJ file. Consult TAF data base administrator.	TAF
FATAL CIO ERROR STATUS.	A TAF CIO operation returned a fatal error status which aborted TAF.	Inform site analyst.	TAF
FIELD LENGTH EXCEEDED FOR CMM.	TAF does not have enough field length to allocate the space potentially required by CMM.	Increase TAF initialization field length.	TAF
FIELD LENGTH EXCEEDED FOR LOCKS.	TAF does not have enough initialization field length for allocating lock tables.	Decrease the locks parameter on the CRM statement, increase the TAF initialization field length, or inform site analyst.	TAF
FIELD LENGTH EXCEEDED FOR RECORD.	TAF does not have enough field length to allocate the space for the record buffer.	Decrease the record size specified in the xxJ file or increase the TAF initialization field length.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
FIELD LENGTH EXCEEDED FOR USERS.	TAF does not have enough initialization field length for allocating file control tables.	Decrease the users parameter on the CRM statement, increase the TAF initialization field length, or inform site analyst.	TAF
FILE edt EMPTY.	Element descriptor table file edt is empty.	Correct error, reinitialize executive, and rerun.	TAF BDMI
FILE xxJ NOT FOUND.	Transaction subsystem aborts. Data base in TCF file has no xxJ file, or a PFM error occurred.	Consult TAF data base administrator or contact site analyst.	TAF
FILE NAME CONFLICT.	The input file name specified on the KTSDMP control statement is the same as the output file name specified.	Correct error and rerun.	KTSDMP
FILE NAME MUST BE 2-7 CHARACTERS.	The xxpfni parameter on the CRM statement must be two to seven characters, the first two (xx) being the data base name.	Correct the xxpfni parameter on the CRM statement or inform site analyst.	TAF
FILE NOT ATTACHED - filename.	Either the P or the N parameter file was not attached when the DA option was used; filename is the permanent file name.	Ensure that the file is defined or attached prior to entering the LIBTASK statement.	LIBTASK
FILE hash NOT FOUND.	The indirect file named hash containing the binary code of the hashing routine was not found under the usernam parameter on the USER statement in the xxJ file or a PFM error occurred.	Ensure that file hash is saved under the usernam parameter or inform site analyst. Consult the CYBER Loader Reference Manual (listed in the preface).	TAF
FILE TCF EMPTY.	An empty TCF exists under the TAF user name.	Place the necessary information on TCF.	TAF
FILE TCF NOT FOUND.	The TCF was not found under the user name of the Transaction Facility.	Create a TCF file under the TAF user name.	TAF
FILE edt TOO LARGE.	Element descriptor table is too large to be read into memory.	Correct error and reinitialize executive (TAF), or rerun (BDMI).	TAF BDMI
FILE TYPE MUST BE IS OR DA.	The type parameter on the CRM statement must be either IS (indexed sequential) or DA (direct access).	Correct the type parameter on the CRM statement or inform site analyst.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
FL REQUEST BEYOND MFL (CM).	CM field length requirements for the task exceed the CM field length allowed.	Contact data base administrator.	MSABT
FL TOO LARGE- nnnnnnB,tasknam,tasklib.	The initial load field length, nnnnnnB, for task tasknam on task library tasklib exceeds the minimum size of the transient task area (potential space available to contain transient tasks). Thus a situation could arise in which it would not be possible to load the task.	Correct error.	TAF
FORMAT ERROR IN THE NETWORK DESCRIPTION FILE.	During transaction executive initialization, one or more errors were found to exist in the network description file.	Inform site analyst.	TAF
FWA .GE. LWA+1.	There is a logical error in the structure of the input file which implies that the first word address is greater than or equal to the last word address plus one.	Inform site analyst.	KTSDMP
HST NOT AVAILABLE.	NAM is not communicating with the 255x communications processors. Either NAM was not initialized or has since failed.	Initialize NAM if it was not initialized previously; inform site analyst if NAM was active but a malfunction occurred.	TAF
ILLEGAL APPLICATION, TRY AGAIN.	The requested application is unknown or unavailable to the user.	Ensure the accuracy of the entry. If problems persist, contact installation personnel concerning validations.	NETVAL
ILLEGAL COMMON MEMORY MANAGER REQUEST.	Memory request with reserved bits in the parameter block set incorrectly.	Do not set reserved bits.	MSABT
ILLEGAL DATA BASE IN xxJ FILE.	One of the statements in the xxJ file specifies an incorrect xx parameter and causes the transaction subsystem to abort.	Examine xxJ files. Consult the TAF data base administrator.	TAF
ILLEGAL FWA FOR LOAD.	The FWA of the load was not equal to 111B. Task binaries must be in absolute format.	Ensure that the correct load procedure is performed.	LIBTASK
ILLEGAL JOURNAL FILE REQUEST.	Either a task requested that an entry be made on an undefined journal file, or the task attempted to make an entry longer than 2499 CM words on a journal file.	Inform site analyst.	MSABT
ILLEGAL LIBTASK ATTEMPT - filename, username.	The transaction executive validates all dynamic attempts to change the task library by comparing the user number of the list of the requester against data base user numbers. If it does not match, or if the	Correct and reinitialize transaction executive.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
	library file is not attached by TAF, the transaction executive issues this dayfile message, where usernum is the user number of the illegal attempt.		
ILLEGAL NUMBER FOR LOCKS.	The locks parameter on the CRM statement is in error. One of the following format conditions exists. - A nonnumeric character. - A character after a postradix of B or D. - An 8 or 9 with a postradix of B.	Correct the locks parameter on the CRM statement or inform site analyst.	TAF
ILLEGAL NUMBER FOR USERS.	The users parameter on the CRM statement is in error. One of the following format conditions exists. - A nonnumeric character. - A character after a postradix of B or D. - An 8 or 9 with a postradix of B.	Correct the users parameter on the CRM statement or inform site analyst.	TAF
ILLEGAL PARAMETER IN RA+1 CALL.	A task error. The parameters specified in a request from a task are either insufficient or invalid.	Correct task or inform site analyst.	MSABT
ILLEGAL REDUCTION OF FL.	Task request to reduce FL would cause the last word address of the communication block to be outside the task FL.	Contact data base administrator.	MSABT
ILLEGAL SUBTRANSACTION CODE - INITIAL TASK.	There is an unrecognizable subtransaction code in the terminal input.	Correct terminal input or inform site analyst.	MSABT
ILLEGAL TASK NAME REQUESTED FOR SCHEDULING.	The task requested was not found in the task library directory.	Correct task or inform site analyst.	MSABT
ILLEGAL TERMINAL INPUT REQUEST.	A task error. A WAITINP request was issued by a memory resident task or by a task that did not originate from a terminal.	Correct task or inform site analyst.	MSABT
ILLEGAL TERMINAL NAME.	A batch job submitted a transaction specifying a nonexistent terminal and/or user name.	Correct task or correct and reinitialize transaction executive with terminal and user name defined.	TAF
ILLEGAL TOTAL INTERLOCK REQUEST.	One of the following. - A task that is not on the system task library issued a CHKON request. - A task issued a CHKON request with a CHKON request already in effect. - A task issued a CHKOFF request without first issuing a CHKON request. - A task that issued a CHKOFF request had a sequence number different from a task that issued a CHKON request.	Inform site analyst.	MSABT

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
ILLEGAL USER ACCESS.	The user tried to perform an operation for which he is not validated. Possible causes include attempts to - run a system origin job from nonsystem origin - access a restricted subsystem without proper validation - enter an invalid SRU value - use the V carriage control character without validation	Ensure accuracy of control statement or determine proper validation requirements via LIMITS statement.	LFM MSI NETVAL QFSP RESEX IMA
ILLEGAL USER ACCESS - CONTACT SITE OPR.	The security count for the user number specified has been decremented to zero. Therefore, the user is denied all access to the operating system until the operator resets the security count.	Contact site personnel to reestablish access.	CPM NETVAL IAJ ILS
ILLEGAL WORD COUNT ON TERMINAL OUTPUT.	One or more of the following task errors occurred on a SEND request. - The word count was less than or equal to zero. - There was an attempt to use SEND MAXWS or more CM words of information in one SEND; MAXWS is an installation option. - There was an attempt to use SEND MAXTO or more CM words of information by one task through a series of SEND requests. MAXTO is an installation option.	Correct task or inform site analyst.	MSABT
INACTIVE TASK REQUESTED.	An attempt was made to schedule a task which has been turned off with a K.OFFTASK command or a LIBTASK directive.	Correct condition or inform site analyst.	OFFTASK
INITIAL TASK NOT IN TASK LIBRARY DIRECTORY.	The task library file does not contain the initial task (ITASK).	Inform site analyst.	TAF
INITIALIZATION OPTIONS.	This message precedes messages indicating the values of the initial K-display options either during initialization or recovery.	None.	TAF
INPUT RESUMED.	The system overload condition has cleared.	Resume terminal input.	CCP
INPUT STOPPED message.	Because of a system overload condition, the network cannot accept terminal input. The message field is an installation-defined text string.	Wait until the message INPUT RESUMED is issued before proceeding.	CCP
INSUFFICIENT FL FOR DATA MANAGER.	The transaction executive requires more field length at initialization time than is available.	Correct error and reinitialize executive.	TAF

MESSAGE	SIGNIFICANCE	ACTION	ROUTINE
INVALID DATA BASE NAME ON DMS STATEMENT.	A data base name associated with TAF, CRM, or OTHER exceeds two characters.	Correct the DMS statement on TCF file.	TAF
INVALID TASK NAME.	The error message was issued for one of the following reasons: - Initialization of a task by the name of ITASK, KDIS, MSABT, OFFTASK, SYSMSG, or XTASK was attempted using XTASK. These are names of required system tasks and will not be initiated by XTASK. - A zero length task name was specified. - A task name of more than seven characters was specified.	Correct the task name.	XTASK
INVALID TCF ENTRY.	A syntax error was detected on the TCF file. This could include one of the following: - The data manager identifier is not TAF, CRM, TOTAL or OTHER. - A data manager status declaration is not in the proper form; it must be ON or OFF.	Correct error.	TAF
INVALID TOTAL PARAMETER LIST.	Parameter of Total is outside the field length.	Inform site analyst.	MSABT
JOURNAL TYPE DOES NOT MATCH xxJ FILE.	Journal file entries in the xxJ file do not match the files themselves. This causes the transaction subsystem to abort.	Consult TAF data base administrator. Examine xxJ file for the journal file entries.	TAF
K.CMB=nn. K.ECS=nnnK. K.MDM=n. K.MFL=nnnnnnB. K.REC=a. K.SCP=nn. K.TLF=a.	Values of the initial K-display options at either initialization or recovery.	None.	TAF
K-DISPLAY COMMAND UNSUCCESSFUL, FORMAT ERROR--command.	There is an error in the syntax of the command or in one of the parameters. The command field is the attempted command or blanks if the command is longer than seven characters or contains an illegal character.	Correct the command and retry the operation.	SYSMSG
K-DISPLAY COMMAND UNSUCCESSFUL, ILLEGAL COMMAND--command.	The command is not a valid K-display command or was misspelled. The command field is the attempted command.	Correct the command and retry the operation.	SYSMSG
K-DISPLAY COMMAND UNSUCCESSFUL, SYSTEM BUSY--command.	The system is busy and cannot respond to the request. The command field is the attempted command.	Retry the operation.	SYSMSG

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
K-DISPLAY COMMAND UNSUCCESSFUL, DROP IGNORED--command.	A K.DROP or K.DDROP could not be processed because the task was in recall, the time-sharing executive denied a request, or a command was issued during the initial load of the task. The command field is the attempted command or request; blanks indicate no command was issued.	Reenter the K.DROP or K.DDROP command. When the recall operation, time-sharing request, or initial load operation is completed, TAF accepts the command and aborts the task.	SYMSMSG
K.MAXFL,nnnnnnB.	The run-time K-display command K.MAXFL was entered with the indicated value.	None.	TAF
K.MAXFL REJECTED.	A value was entered which caused potential blocked tasks to be detected.	Reenter K.MAXFL with a larger value.	TAF
KTSDMP COMPLETE.	Dump completed normally.	None.	KTSDMP
LIBRARY DIRECTORY EMPTY - filename.	The task library file indicated does not contain a directory.	Inform site analyst.	TAF
LIBRARY DIRECTORY ERROR - filename.	The task library file indicated contains a nonrecognizable directory.	Inform site analyst.	TAF
LIBRARY DIRECTORY TOO LONG - filename.	The directory record on the task library file indicated exceeded the maximum length allowed by the transaction executive (398 entries).	Inform site analyst.	TAF
LIBRARY FILE TIME OUT.	The library file was busy for the length of time specified by an assembly constant, so LIBTASK was unable to attach it. Another user apparently has the file attached and will not allow modification.	Retry at a later time.	LIBTASK
LIBTASK-PFM ERROR DETECTED.	A permanent file error other than FILE BUSY was received when the library file attach was attempted.	Correct and rerun.	LIBTASK
LOADING ECS tasknam.	Informative message. The transaction subsystem is loading task tasknam.	None.	TAF
LOGIN ABORTED, TRY LATER.	Insufficient resources are available to allow the user to gain access to the network.	None.	NETVAL
MEMORY OVERFLOW DURING INITIALIZATION.	TAF aborted because its field length for initialization was insufficient.	Inform site analyst. IFL= in deck TAF should be increased. Increasing the central memory field length parameter on the RFL control statement in the TAF initialization procedure file (ffff) does not correct this problem.	TAF

MESSAGE	SIGNIFICANCE	ACTION	ROUTINE
MEMORY REQUEST FOR ECS NOT ALLOWED.	Illegal request of ECS by task.	Contact data base administrator.	MSABT
MEMORY REQUEST NOT ALLOWED WITH DATA MANAGER REQUESTS OUTSTANDING.	A task error. MEM or RFL request attempted with data manager requests outstanding.	Correct task or inform site analyst.	MSABT
MFL TOO LARGE - nnnnnnB,tasknam,tasklib.	The MFL (initial field length plus expandable field length) of the non-CM resident task (tasknam) on task library (tasklib) exceeds the minimum size of the transient task area (potential space available to contain transient tasks). Thus a situation could arise in which it would not be possible to complete processing of this task.	Reduce the task FL or EF, or increase the TAF FL.	TAF
MINIMUM TAF MFL NEEDED = nnnnnnB.	Potentially blocked tasks were detected at one of the following times: - TAF initialization - Attempted task library update - Attempt to change TAF maximum FL via K.MAXFL command. The above operation did not complete normally. The maximum FL of TAF must be at least nnnnnnB. If nnnnnnB exceeds the largest field length possible for TAF (377700B), then other corrective action is needed.	Correct error.	TAF
MISSING HEADER WORD ON xxJ FILE.	The first statement on the xxJ file is in error, causing the transaction subsystem to abort.	Examine xxJ files for header xxJ. Consult the TAF data base administrator.	TAF
MORE THAN FIVE TASKS IN TASK CHAIN.	A task error. Only five tasks can be requested for scheduling in one CALLRTN, CALLTSK, or NEWTRAN request; more than five were requested.	Correct task or inform site analyst.	MSABT
MORE THAN ONE ENTRY POINT ON A *54* TABLE.	Multiple entry points are not allowed in tasks.	Rewrite the task with only one entry point.	LIBTASK
NAM ERROR - ILLEGAL ABH.	The application block header (ABH) sent to TAF by NAM is unrecognizable.	Inform site analyst.	TAF
NAM ERROR - ILLOGICAL ABT.	The application block type (ABT) sent to TAF by NAM is unrecognizable.	Inform site analyst and refer to the NAM Reference Manual.	TAF
NAM FUNCTION NOT FOUND.	TAF received a supervisory message from NAM which had an unrecognizable primary or secondary function code.	Perform a dump of TAF and NAM or inform site analyst.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
NAM LOGICAL ERROR.	NAM sent TAF a message out of order or an unrecognizable message.	Inform site analyst.	TAF
NAM NOT AVAILABLE.	Informative message indicating that TAF is currently at a control point but NAM is not. Transactions can be initiated from batch only.	Bring NAM to a control point, if desired.	TAF
NAM PHYSICAL ERROR EC=ec.	NAM has detected a physical error indicated by error code ec.	Refer to the NAM Reference Manual for the meaning of this error code.	TAF
NAM REJECT.	During login processing, NAM rejected the terminal.	Inform site analyst.	TAF
NEED AT LEAST xx SUBCONTROL POINTS.	There are more CM resident tasks defined than subcontrol points. If non-CM resident tasks exist, there must be at least one more subcontrol point than there are CM resident tasks.	Reinitialize the transaction executive and assign more subcontrol points, or reduce the number of CM resident tasks.	TAF
NETOFF COMPLETE.	Informative message indicating that TAF is no longer communicating with NAM. NAM initiated shutdown procedures prior to loss of communications.	When NAM is available, the central site console operator command K.NAMON can be used to resume communications between TAF and NAM.	TAF
NETON COMPLETE.	Informative message indicating that TAF is communicating with NAM.	None.	TAF
NETWORK DESCRIPTION FILE NOT FOUND.	The NCTFid file is not present under the system user index for terminal communications.	Inform site analyst.	TAF
NETWORK SHUT DOWN DETECTED.	Self-explanatory.	None.	TAF
NO ACCOUNT/USER CARD IN xxJ FILE.	The ACCOUNT or USER statement in the xxJ file is not present, causing the transaction subsystem to abort.	Add ACCOUNT or USER statement in xxJ file. Consult the TAF data base administrator.	TAF
NO CRAS TERMINAL DEFINED.	Informative message indicating that no CRAS terminal is defined in the terminal validation file.	None.	TAF
NO DATA BASE ID FOR DATA MANAGER.	At least one data base identifier must be specified on each active (ON) DMS statement.	Add data base identifier to DMS statement(s) or specify status as OFF.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
NO DATA BASE NAME IN xxJ FOR TOTAL.	Self-explanatory.	Add data base name to xxJ file.	TAF
NOT AVAILABLE.	K-display message indicating that the entry is not available for this version.	None.	TAF
NULL DESCRIPTION FILE.	Self-explanatory.	Create a description file (NETWid, SIMFid, or NCTFid where id is the machine identifier).	TAF
OFF TASK tasknam-LIBRARY libnam.	Task tasknam in task library libnam could not be loaded from ECS or recovered and loaded from mass storage. Task was turned off. Transactions using tasks will abort.	Inform site analyst. Library must be recreated.	TAF
OPERATOR IDLE DOWN.	The operator dropped TAF.	None.	TAF
OPERATOR STOP.	Informative message indicating that the operator has stopped TAF.	None.	TAF
POOL FILE - filename NOT FOUND IN EDT.	The pool file xxERPF contains a record for file filename not defined in the EDT for data base xx. A CRAT entry is not made for the record for this file.	Inform site analyst.	BDMI TAF
POSSIBLE BLOCKAGE AMONG CM RESIDENT TASKS.	The sum of the maximum field lengths (MFLs) for the CM resident tasks exceeds the minimum size of the total task area (potential space available to contain tasks). Thus one or more CM resident tasks could be blocked from completing.	Correct error.	TAF
POTENTIALLY BLOCKED TASKS DETECTED.	During TAF initialization, potentially blocked tasks were detected. Preceding error messages contain additional details.	Correct error.	TAF
RA+1 CALL ERROR.	There was an unrecognizable task request.	Correct task or inform site analyst.	MSABT
RA+1 CALL PARAMETER ADDRESS OUTSIDE FL.	A parameter in a task request was outside the field length of the task.	Correct task or inform site analyst.	MSABT
RA+1 CALL WITH ILLEGAL FUNCTION CODE.	There was an unrecognizable task request for system action.	Correct task or inform site analyst.	MSABT
RECOVERY COMPLETE.	The transaction executive or time-sharing subsystem has successfully completed recovery.	None.	TAFNAM2 TAFTS2 IAFEX TELEX

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
RECOVERY IMPOSSIBLE.	A fatal error occurred during transaction executive initialization or the recovery file prepared by the TAF initialization routine does not exist or could not be read.	Check dayfile for a specific error message or inform the site analyst.	TAFNAM2 TAFTS2
RECOVERY IN PROGRESS.	The time-sharing subsystem or the transaction executive has begun recovery procedures due to an abort or termination condition.	None.	TAFNAM2 TAFTS2 IAFEX TELEX
REFERENCE MADE TO ILLEGAL TERMINAL NAME.	The terminal name or user name specified has not been defined.	Correct task or inform site analyst.	MSABT
REPEAT..	Because of a temporary overload condition, the network has discarded the last logical line of input.	Reenter the information.	CCP NETVAL
REQUEST NOT ALLOWED WITH DATA MANAGER REQUESTS OUTSTANDING.	A task error. CALLRTN, WAIT or WAITINP request was attempted with data manager request(s) outstanding.	Correct task or inform site analyst.	MSABT
REQUESTED ECS NOT AVAILABLE.	The amount of ECS requested was not available in a contiguous block.	Reinitialize with less ECS requested.	TAF
SEND TO TERMINAL NOT LOGGED IN.	A terminal issued a SEND without recall and one of the following occurred. - TAF was not connected to NAM. - The terminal to which the SEND was attempted was not logged into TAF. - TAF received a STOP supervisory message for the destination terminal.	Correct error condition or inform site analyst.	MSABT
SYSTEM ABORT OF TASK.	The operator aborted the task (via a K.DROP or K.DDROP command), or a data manager error caused the task to abort.	Inform site analyst.	MSABT
SYSTEM CLOSED.	The system is not available for use by terminals.	None.	NETVAL IAFEX TELEX
TABLE OVERFLOW-TOO MANY TASKS.	The managed tables used by LIBTASK are not large enough for the number of tasks or directives.	LIBTASK must be reassembled with larger table allocation.	LIBTASK
TAF DATA MANAGER SUCCESSFULLY LOADED.	Self-explanatory.	None.	TAF
TAF INTERNAL ERROR.	TAF has found internal data to be inconsistent.	Perform a dump of TAF or inform site analyst.	TAF
TAF NETON COMPLETE.	TAF is connected to the network.	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
TAF NOT CALLED BY DSD.	The transaction subsystem aborts if initiated in any way other than DSD command.	None.	TAF
TAFNAM/TAFTS TERMINATE.	Transaction subsystem has terminated.	None.	TAF
TASK EDITING COMPLETE.	Dayfile message indicating that the run has terminated. Normal LIBTASK output consists of four lists - the input directives, the tasks on the current library, the tasks added to the library, and the tasks replaced on the library. After these lists, the time and date of the last library modification are given. If errors occurred during processing of the task binary file, a list of error messages also appears. If the LIBTASK run was initiated from a terminal, the directive and current library lists do not appear.	Refer to next dayfile message.	LIBTASK
TASK LIBRARY DIRECTORY EMPTY.	The last record of the library file is empty.	Ensure that the correct file has been specified; if so, the library must be recreated.	LIBTASK
TASK LIBRARY DIRECTORY EMPTY - libnam.	The file specified as the task library contains no recognizable directory. TAF aborts.	Inform site analyst. Task library libnam must be corrected and TAF reinitialized.	TAF
TASK LIBRARY DIRECTORY ERROR - libnam.	Task library libnam contains no recognizable directory. TAF aborts.	Inform site analyst. Task library must be corrected.	TAF
TASK LIBRARY DIRECTORY ERRORS.	The first word of the last record on the library file does not contain the directory header.	Correct and rerun.	LIBTASK
TASK LIBRARY DIRECTORY TOO LONG.	The directory is too long for the buffer allocated.	This may be altered by changing an assembly constant.	LIBTASK
TASK LIBRARY DIRECTORY TOO LONG - libnam.	The directory record on task library libnam exceeded the maximum length allowed by the transaction executive (170 files). TAF aborts.	Inform site analyst. Size of task library libnam must be reduced and TAF reassembled.	TAF
TASK NOT VALIDATED FOR REQUEST.	One of the following actions has occurred. - The terminal operator initiated a transaction which tried to perform an action associated with a data base for which the terminal was not validated. - A NEWTRAN request was issued by a task not in the system task library (TASKLIB).	Perform the appropriate action. - Inform site analyst; transaction must be reinitialized. Set up the terminal name in the network file to use the data	MSABT

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
		base. The system data base (SY) may be used. - Put the task on TASKLIB.	
TASK REQUEST ARGUMENT ERROR.	A parameter error occurred in a request.	Correct task or inform site analyst.	MSABT
TASK REQUESTED CEASE BEFORE DM FINISHED.	The task requested to end before the data manager completed all outstanding requests.	Correct task or inform site analyst.	MSABT
TASK TIME LIMIT.	The task exceeded n time slices. The length of the time slice is set by an installation parameter. The number of time slices, n, allowed for the execution of a task can be changed with a task ITL request.	Increase time slices or inform site analyst.	MSABT
TERMINALS MISSING IN NETWORK FILE.	A valid network file was found but no trans-action terminals were defined in it. If TAF interfacing with TELEX is being used, TELEX was not reinitialized to recognize the same network file that TAF was using.	Ensure that the network file is under the proper name; if it is, inform site analyst. If TAF interfacing with NAM is being used, the network file is NCTFid. If TAF interfacing with TELEX is being used, the network file is NETWid if TAF/TS can attach it; otherwise, TAF/TS uses SIMFid. For the TELEX interface, reinitialize TELEX and TAF. For the NAM interface, reinitialize TAF.	TAF
TIMEOUT.	The user has not responded to a login prompt within the system-specified time interval (usually about 2 minutes) and the terminal was disconnected.	Respond more quickly to prompts.	NETVAL
TLD OVERFLOW.	Space is reserved at initialization to dynamically add tasks to the TAF resident copy of each task directory. The number of tasks is limited (default is 10) by the installation parameter TLDL. This message indicates that more tasks than allowed have been added since the most recent TAF initialization. The extra tasks will be ignored by TAF until it is reinitialized.	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
TOO MANY BRANCHES IN TASK CHAIN REQUESTED.	Only three branches are allowed in the task chain; more than three were requested.	Correct task or inform site analyst.	MSABT
TOO MANY CRAT FILE ENTRIES.	There are too many error records on the xxERPF file.	Inform site analyst. The bad areas on disk should be flawed and the data base reloaded.	TAF
TOO MANY DATA BASE NAMES.	The number of data base names associated with one data manager via DMS statements exceeds the value of MAXDB.	Decrease the number of data base names associated with the data manager.	TAF
TOO MANY FILES IN TOTAL DATA BASE.	Self-explanatory.	Reduce the number of entries in the TCF file or increase TMAXFIL.	TAF
TOO MANY JOURNAL FILES IN xxJ FILE.	More than three journal files per data base were specified, causing the transaction subsystem to abort.	Examine xxJ file for xxJOR entries. Consult the TAF data base administrator.	TAF
TOO MANY RA+1 CALLS.	Only n RA+1 requests are allowed; more than n occurred, where n is an assembly constant.	Inform site analyst. Possibly a task is in a loop.	MSABT
TOTAL DATA MANAGER NOT LOADED.	Total data manager was called but has not been loaded.	Inform site analyst.	MSABT
TOTAL DATA MANAGER SUCCESSFULLY LOADED.	Self-explanatory.	None.	TAF
TOTAL DID NOT RECOVER PROPERLY. STATUS IS yyyy.	An error status yyyy was returned on a Total FINAL call. Refer to Diagnostics in the Total Reference Manual for yyyy.	Correct error and reinitialize transaction executive.	TAF TAFNAM2 TAFTS2
TRN - ABNORMAL.	A TAF-originated task could not be scheduled because of a lack of communication blocks or ITASK queue was full. An attempt to schedule the task will be made at a later time.	Inform site analyst.	TAF
TT OPTION REQUIRES USER NUMBER.	When updating a task library on-line (TT option is specified on LIBTASK statement), the user number must be specified prior to the LIBTASK statement so the library associated with that user number can be found.	Specify user number via USER or ACCOUNT statement before LIBTASK statement and rerun job.	LIBTASK
UN=userid NOT VALID ON FM=family.	The transaction subsystem initialization aborted. User name userid is not validated for use on the specified family.	Consult data base administrator.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
UNABLE TO ATTACH TOTAL BINARIES.	File of Total binaries is not under the user index of the transaction subsystem or a PFM error occurred.	Correct error and reinitialize transaction executive or contact site analyst.	TAF
UNABLE TO ATTACH TOTAL DBMOD BINARIES.	One or more of the DBMOD files listed on the TOTAL DMS control statement in the TCF file could not be attached under the user index of the transaction subsystem or a PFM error occurred.	Correct error and reinitialize transaction executive or contact site analyst.	TAF
UNKNOWN FILE FORMAT.	There is a logical error in the structure of the input file. It does not conform to the established format rules.	None.	KTSDMP
UNKNOWN FORMAT ON ENTRY POINT TABLE.	The task binary file contains records in the wrong format.	None.	LIBTASK
UNRECOGNIZABLE CIO ERROR CODE.	Self-explanatory.	Inform site analyst.	TAF
WAITING FOR NAM.	NAM is not operative.	Inform site analyst to initialize NAM.	NETVAL
WAITINP FROM MULTI QUEUED TASK.	A WAITINP request was issued by a task that was scheduled to process more than one transaction or by a task that was called with return.	Correct task or inform site analyst.	MSABT
n.nnn AVERAGE ACTIVE SUBCONTROL POINTS.	Average number of simultaneously active subcontrol points when TAF is not rolled out. An active subcontrol point is one which is in recall, is waiting to use the CPU, or is currently assigned the CPU. The sampling rate is once per second.	None.	TAF
n.nnn AVERAGE OUTSTANDING CDCS REQUESTS.	Average number of outstanding (uncompleted) SSC requests per second. The sampling rate is once per second.	None.	TAF
n.nnn KILO CDCS REQUEST REJECTS FOR BUSY.	Total number of SSC rejects for busy when less than seven outstanding CDCS SSC requests existed at the time of the current request.	None.	TAF
n.nnn KILO CDCS REQUEST REJECTS FOR MAXR.	Total number of SSC attempts when there were seven (MAXR) outstanding CDCS SSC requests.	None.	TAF
n.nnn KILO CDCS REQUESTS FROM TASKS.	Total number of CDCS SSC requests issued by tasks. The number does not include terminate requests which are blocked by TAF.	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
n.nnn KILO ITASK RELOADS.	Upon transaction facility termination, this message indicates how many reloads of ITASK occurred. Because ITASK is a central memory resident task, if it makes a fatal error, it must be reloaded.	Inform site analyst.	TAF
n.nnn KILO LOCK REJECTS-filenam.	TAF CRM could not lock records in filenam for a task because the locks parameter on the CRM statement for that file was not large enough.	The site analyst should consider increasing the locks parameter on the CRM statement if lock failures occur frequently.	TAF
n.nnn KILO LOCKS-filenam.	TAF CRM processed nnnn locks for file filenam.	None.	TAF
n.nnn KILO NO COMMUNICATION BLOCKS.	Upon transaction facility termination, this message indicates the number of times a communication block was needed and was not available.	Inform site analyst. The site analyst might want to consider using the K.CMB command during TAF initialization.	TAF
n.nnn KILO NO FL FOR TASK LOAD.	Upon transaction facility termination, this message indicates the number of times TAF was unable to obtain enough memory to load tasks.	Inform site analyst. The site analyst might want to consider using the K.MFL command during TAF initialization.	TAF
n.nnn KILO NO SUBCONTROL POINTS.	Upon transaction facility termination, this message indicates the number of times a subcontrol point area was needed and was not available.	Inform site analyst. The site analyst might want to consider using the K.SCP command during TAF initialization.	TAF
n.nnn KILO OPEN REJECTS-filenam.	TAF CRM could not open file filenam for a task because the users parameter on the CRM statement is not large enough.	The site analyst should consider increasing the users parameter on the CRM statement if open failures occur frequently.	TAF
n.nnn KILO OPENS-filenam.	TAF CRM processed nnnn opens for file filenam.	None	TAF
n.nnn KILO STORAGE MOVES OF TASKS.	Upon transaction facility termination, this message indicates the number of subcontrol points that were moved. This is done to make more effective use of the field length of TAF.	None.	TAF
n.nnn KILO TAF FL INCREASES.	Upon transaction facility termination, this message indicates the number of times TAF increased its field length while trying to satisfy an internal request for a block of	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
	memory. This internal request is often issued to obtain memory in which to load a task.		
n.nnn KILO TASK RECALLS.	Upon transaction facility termination, this message indicates the number of times tasks were put in recall. This is an internal statistic over which the applications programmer has little control.	None.	TAF
n.nnn KILO TASK RECALLS FOR OUTPUT.	Upon transaction facility termination, this message indicates the number of times tasks were put in recall because the network block limit (NBL parameter on a TERMINAL statement) for the terminal in use was reached.	Refer to the Network Access Method Network Definition Language Reference Manual (listed in the preface).	TAF
n.nnn KILO TASK RELOADS.	Upon transaction facility termination, this message indicates how many reloads of tasks occurred. TAF reloads tasks that have fatal errors if one of the following is true. - The task is central memory resident. - There are queued transactions waiting to use the task.	Inform site analyst.	TAF
n.nnn KILO TASK ROLLOUT COMPLETES.	Upon transaction facility termination, this message indicates the number of times TAF rolled out tasks. This statistic excludes the number of times the rollout operation did not complete.	None.	TAF
n.nnn KILO TASK ROLLOUT STARTS.	Upon transaction facility termination, this message indicates the number of times TAF initiated a rollout operation. Rollout includes writing a task to disk and releasing the task subcontrol point space. In some cases, the task is allowed to stay central-memory-resident for a while before it is rolled out.	None.	TAF
n.nnn KILO TASK ROLLOUTS BY TAF DM.	Upon transaction facility termination, this message indicates the number of times a rollout was initiated for a task as a result of the TAF data manager detecting a locked record. This occurs when the TAF data manager is unable to proceed because a record has been locked for a certain amount of time. The rollouts are initiated by a request from the TAF data manager.	None.	TAF
n.nnn KILO TASK STARTS BY TAF D.M.	Upon transaction facility termination, this message indicates the number of times tasks were activated as a result of a TAF data manager event, such as a locked record.	None.	TAF

<u>MESSAGE</u>	<u>SIGNIFICANCE</u>	<u>ACTION</u>	<u>ROUTINE</u>
n.nnn KILO TRANSACTION ABORTS.	Upon transaction termination, this message indicates how many transaction tasks have aborted.	Data base administrator may have to correct data base to account for transactions.	TAFNAM2 TAFTS2
n.nnn KILO TRANSACTIONS PROCESSED.	Self-explanatory.	None.	TAFNAM2 TAFTS2
filenam NOT FOUND.	The user specified the P option on the KTSDMP control statement; one of the two files was not in the user's permanent file catalog or a PFM error occurred.	Specify a different file name or inform a site analyst.	KTSDMP
xxxxxx NOT INITIALIZED BY TOTAL. STATUS IS yyyy.	An error was encountered on the Total data base.	Regenerate Total data base. Refer to Total Reference Manual for status.	TAF
nnnn PER CENT CPU USAGE.	Summary message indicating CPU usage by the transaction subsystem.	None.	TAFNAM2 TAFTS2
nnn TASKS NOT LOADED INTO ECS.	An insufficient amount of ECS was available to load all tasks. The nnn field is the number of tasks not loaded.	Check ECS requested and reinitialize with more ECS if appropriate.	TAF

The sample deck structures in this appendix demonstrate the use of various parameters which are available. The default input and output files can be used to reduce the number of control language statements needed to bring about a compile, load, and LIBTASK sequence, because the default binary output files are the binary input files for subsequent processors.

For example:

```
COBOL5,ANSI=77LEFT,UC1.
LOAD(LGO)
NOGO(LGOB)
LIBTASK,P=RLTASKL,TT.
```

It is assumed that each of the following decks is preceded by job and USER statements. Some sites may require a CHARGE statement.

COMPASS TASK DECK STRUCTURE

A sample deck which assembles a COMPASS application program and loads it onto a task library, RLTASKL, is illustrated in figure C-1.

```
ATTACH,OPL/UN=username.
COMPASS.
LDSET(LIB=TRANLIB)
LOAD(LGO)
NOGO(LGOB)
LIBTASK(P=RLTASKL,B=LGOB,TT,I=0)
7/8/9
    IDENT  TASK
    ENTRY  TASK
OPL  XTEXT  COMKMAC
COMBL BSS    69
    .
    . ← COMPASS task appears here
    .
END
6/7/8/9
```

Figure C-1. COMPASS Deck Structure

NOTE

The system OPL must be attached for COMKMAC to be called in the COMPASS program. Consult site personnel for the appropriate control statement.

FORTRAN TASK DECK STRUCTURE

A sample deck which compiles a FORTRAN application program and loads it onto a task library, RLTASKL, is illustrated in figure C-2.

```
FTN.†
LDSET(LIB=TRANLIB)
LOAD(LGO)
NOGO(LGOB)
ATTACH(RLTASKL/M=W)
LIBTASK(P=RLTASKL,TT,I=0)
7/8/9††
OVERLAY(ABS,0,0)
PROGRAM TASK2
.
. ← FORTRAN task appears here
.
6/7/8/9
```

†The SEQ, TS, and OPT=0 options are not allowed.
††TASK2 is the name of the task.

Figure C-2. FORTRAN Deck Structure

COBOL TASK DECK STRUCTURE

A sample deck which compiles a COBOL application program and loads it onto a task library, RLTASKL, is illustrated in figure C-3.

```
COBOL5,ANSI=77LEFT,UC1.
LDSET(LIB=TRANLIB)
LOAD(LGO)
NOGO(LGOB)
LIBTASK(P=RLTASKL,B=LGOB,TT,I=0)
7/8/9
.
. ← COBOL task appears here
.
6/7/8/9
```

Figure C-3. COBOL Deck Structure

NOTE

Under COBOL, the PROGRAM-ID name is used for the task name. Only alphanumeric names can be used as task names.

Unless otherwise specified in xxJ, all files are assumed to be DI (844) resident.

FILE DESCRIPTIONS

xxJ

This indirect access file identifies the journal files for a specific data base (xx) and provides the user index and user name of the data base.

TASKLIB

System task library. This direct access file contains the application program tasks for a system and is defined under the transaction subsystem user name.

JOUR0

System journal file. The transaction subsystem writes journal entries for all transactions to this direct access file.

xxTFIL

TAF data manager trace file. One direct access trace file for each data base (xx) contains a record of all writes to each file to be traced as a result of transactions. (Refer to the Transaction Facility Version 1 Data Manager Reference Manual.)

xxERPF

TAF data manager error recovery pool file for data base xx. This direct access file contains copies of records which could not be written to their original place of residency because of a mass storage error. (Refer to the Transaction Facility Version 1 Data Manager Reference Manual.)

xxJORN

Optional direct access journal files for each data base (xx) for task journalizing of transactions.

xxTASKL

User-specified task library for data base xx. This must be a direct access file.

TAFNAM

Procedure file containing control statements for TAF using NAM interface.

NCTFid

Terminal network file for declaring users and associated data bases. This must be a direct access file.

TCF

The TAF configuration file. An indirect access file which associates data base names with the desired data manager. It is used to specify which data bases and data managers are currently active (able to be used within TAF).

FILE DEFINITION BY USER NAME

Files used in the transaction subsystem can be divided into three general categories.

- Files defined under the transaction subsystem user name. There is a default user name, with password, and a user index defined for the transaction subsystem. This requires reassembly to change. (Refer to the NOS Installation Handbook for more information.)
- Files defined under a data base user name. These files must be permitted in write or modify mode to the transaction subsystem user name.
- Files defined under system user name.

In the following lists, the letters xx signify the two-character data base name. Files are listed according to the user name under which the files must be defined or are defined by the subsystem.

Associated with Transaction Subsystem User Name

xxJ

TASKLIB

JOUR0

Associated with
Data Base
User Name

xxTFIL†

xxERP†

xxJOR1

xxJOR2

xxJOR3

xxTASKL

xx† (File containing data definition
information for data base xx)

xxaaaa

.

.

xxzzzz

Dxxiiii† (Dual-recorded files for data base xx)

Associated with
System User
Name

TAFNAM

NCTFid

RESERVED FILES

In addition to the files described earlier in this section, the following files are used by TAF. No task or procedure file called by TAF can attach or use any file having one of these names.

KTSROLL

OUTPUT

RECOVR

SCRA-SCRZ

SCR1-SCR5

TROB

MAP

ZZZZZ.. (file names starting with 5 Z's)

†TAF data manager only,

INSTALLATION OVERVIEW

E

Table E-1 is provided to aid the systems analyst and data base administrator in performing TAF installation. It is intended to be a chronologically ordered outline

procedures detailed in the respective manuals. The complete name and publication number of each entry in the Manual column follows the table.

TABLE E-1. INSTALLATION OVERVIEW

Area	Step	Manual	Manual Keyword
1. Install NOS		NOS Install NOS Maintenance	
2. Install Communications Subsystem		NOS Install NOS Maintenance CCP Reference NAM Reference NDL Reference	
3. Install Product Set-COBOL, etc.		NOS Install	COBOL 5 FORTRAN 4 FORTRAN 5 COMPASS
4. Modify/create TAF subsystem	1. Evaluate installation parameters. Some items of interest at initial installation are: • TAF username • TAF user index • TAF password • TAF charge, project numbers • TAF communication interface - 2550's or multiplexers • Number of terminals	NOS Install	Transaction Facility
	2. Modify TAF on deadstart media if required, or	NOS Install NOS Reference	Transaction Facility GTR, LIBEDIT, LIBGEN
	3. Modify TAF on a running NOS system.	NOS Maintenance	SYSEDT
	4. Install TAF username and charge.	NOS Install	TAF installation parameters
	5. Install procedure file to initiate TAF.	NOS Maintenance NOS Install	MODVAL, PROFILE TAF
	6. Install automatic initiation of TAF upon deadstart if desired.	NOS Install	IPRDECK
5. Install Transaction Application	1. Assign a team to do data base design. TAF interfaces to TAF data manager, CRM, and TOTAL. Areas in data base design that require examination include: • Elements to be accessed • Relationships between the elements • Data manager to be used • File, record types, and element types to be used to store the data • Data formats for the tasks • Mass storage of the data base	TAF Reference TOTAL Reference AAM Reference TAF Data Mgr Ref. TAF/CRM Ref. TAF Users Guide CDCS Reference	

TABLE E-1. INSTALLATION OVERVIEW (Contd)

Area	Step	Manual	Manual Keyword
	- Specific processing to be assigned to tasks and the relationships between tasks. For TAF, tasks involved with I/O should be small since the tasks will occupy memory for some time. Tasks that are often used and run quickly may be made core resident. Tasks that are often used should be reusable to avoid task loading - Data base recovery		
	2. Assign a team to do the communications interface. Areas to examine are:		
	Network Access Methods (NAM) features and workings	NAM Reference	
	Communication Control Program (CCP) and 2550 hardware, communication lines, terminals, etc.	CCP Reference	
	Network Description Language (NDL)	NDL Reference	
	TAF terminal I/O requests	TAF Reference	
	Message formats, terminal operator interactions		
	Recovery of terminal communications		
	Interface specifications with the tasks's data management activities		
	NOS and Network Stimulators	NOS Maintenance NPS Reference	
	3. Install a prototype application using the following steps. As you learn more about the capability and performance trade-offs, revise your design. Use stimulators to test the design under a large terminal load.		
	4. Describe and install the data base using the appropriate data manager.		
	CDCS	TAF Reference CDCS Reference	
	CRM	TAF Reference TAF/CRM Reference	XXJ files
	TOTAL	TAF Reference TOTAL Reference	XXJ files
	TAF	TAF Reference TAF Data Manager	XXJ files
	5. Describe and install terminal network files.		
	A. Install TAF's NCTFid file	TAF Reference NOS Maintenance	Network files Network Description files
	B. Install NAM's LCF and NCF Network files	NDL Reference	
	C. Install terminal users in NOS validation files	NOS Maintenance	MODVAL, PROFILE

TABLE E-1. INSTALLATION OVERVIEW (Contd)

Area	Step	Manual	Manual Keyword
	6. Modify procedure file that initializes TAF for special pre-processing procedures, such as preparing for recovery.	NOS Install	TAF
	7. Write application tasks.	TAF Reference	
	• COBOL 5	COBOL 5. Ref. COBOL 5 User's Guide	
	• FORTRAN 4		FORTRAN 4 Ref.
	• FORTRAN 5		FORTRAN 5 Ref.
	• COMPASS		COMPASS Ref.
		NOS Reference Vol. 2	
	8. Modify ITASK to map transaction codes to their associated tasks. Code for handling system transactions and error processing may be desired depending upon application.	TAF Reference NOS Install	ITASK TAF
	9. Modify other TAF furnished system tasks if desired.	TAF Reference	System Tasks
	• LOGT - terminal logout request		
	• MSABT - task		
	• SYSMSG - TAF system messages		
	• Interface specifications with the tasks data management activities		
	10. Build data base task libraries.	TAF Reference	LIBTASK
	11. Build TAF system library.	NOS Install TAF Reference	TAF LIBTASK
	12. Build data base journal files xxJ to give TAF username for data base.	TAF Reference TAF/CRM Reference	xxJ files xxJ files
	13. Build the TAF configuration file (TCF).	TAF Reference	
6. Test the transaction application			
7. TAF operational commands	Initialize TAF from computer console Status TAF from computer console Terminate TAF from computer console	NOS Operator	Transaction Subsystem
8. Modify performance of TAF and application	1. TAF internal design	NOS IMS	TAF
	2. TAF installation parameters	NOS Install	TAF
	3. Terminal stimulations		
	• NPS	NPS Reference	
	• STIMULA	NOS Maintenance	STIMULA
	• NSTIM	NOS Maintenance	NSTIM
	• ASTIM	NOS Maintenance	ASTIM

<u>Manual</u>	<u>Publication Title</u>	<u>Publication Number</u>
AAM Reference	CYBER Record Manager Advanced Access Methods Version 2 Reference Manual	60499300
CCP Reference	Communications Control Program Version 3 Reference Manual	60471400
COBOL 5 Reference	COBOL Version 5 Reference Manual	60497100
CDCS Reference	CYBER Database Control System Version 2 Reference Manual	60481800
COBOL 5 User's Guide	COBOL Version 5 User's Guide	60497200
COMPASS Reference	COMPASS Version 3 Reference Manual	60492600
FORTRAN 4 Reference	FORTRAN Extended Version 4 Reference Manual	60497800
FORTRAN 5 Reference	FORTRAN Version 5 Reference Manual	60481300
NAM Reference	Network Access Method Version 1 Reference Manual	60499500
NDL Reference	Network Access Method Version 1 Network Definition Language Reference Manual	60480000
NOS Install	NOS Version 1 Installation Handbook	60435700
NOS Maintenance	NOS Version 1 System Maintenance Reference Manual	60455380
NOS Operator	NOS Version 1 Operator's Guide	60435600
NOS Reference	NOS Version 1 Reference Manual Volume 1	60435400
	NOS Version 1 Reference Manual Volume 2	60445300

<u>Manual</u>	<u>Publication Title</u>	<u>Publication Number</u>
NPS Reference	Network Products Stimulator Version 1 Reference Manual	60480500
TAF/CRM Reference	Transaction Facility Version 1 CYBER Record Manager Data Manager Reference Manual	60456710
TAF Data Manager Reference	Transaction Facility Version 1 Data Manager Reference Manual	60455350
TAF Reference	Transaction Facility Version 1 Reference Manual	60455340
TAF User's Guide	Transaction Facility User's Guide	60455360
TOTAL Reference	TOTAL - CDC Reference Manual Version 2	76070300

The network TAF uses as a terminal interface can support many types of terminals. All supported terminals are grouped by the network into 15 terminal classes. Each terminal class has terminal operating characteristics associated with it. These are defined by a set of parameters referred to as terminal definitions. These terminal definitions are predefined by the network to match closely the operating characteristics of actual terminals. Table F-1 lists these terminal definitions, their default values, and the ranges (in parentheses) for each of the 15 classes.

When a terminal logs into the network, the network assumes it to be of a certain terminal class or assigns it a terminal class that resembles its actual characteristics. In either case, if the characteristics of the terminal do not match those of its assigned terminal class, the user can change the values of the terminal definitions or even change the terminal class, using the terminal definition commands.

When a terminal definition command is used to change a value, that change remains in effect until the terminal is disconnected from the network or another terminal definition command is used to change the value. Even application switching and TAF disconnection do not change the values of the terminal definitions if the terminal has not been disconnected from the network. Only by disconnecting the phone and redialing (on dial-up terminals) or by entering the TC terminal definition (described later in this section) can the values be reset to their default values.

TERMINAL DEFINITION COMMAND FORMAT

The terminal class or terminal definition values can be changed by entering terminal definition commands in the following format.[†]

control terminal definition **Ⓒ**:

control

Terminal control character defined for the terminal in use (CT character in table F-1).

terminal definition

A two-character mnemonic shown in table F-1, followed by = and the desired terminal class number or parameter value. The range of values is shown in table F-1.

Ⓒ

Message terminator character for the terminal in use.

As an example, to set the cancel character on a CDC 731-12 terminal to a ?, enter:

%CN=? **Ⓒ**

[†]Spaces are included for clarity. They should not be included when entering the command.

On terminals from which it is possible to input multiple logical lines in a single transmission block (more than one line can be entered before transmitting), the terminal definition control character must be the first character in the transmission block. Subsequent input lines in the transmission block are treated as data and sent to TAF. When the terminal definition command is a request for transparent mode input, transparency is not applied to any data in the current transmission block but is applied to the next transmission.

If the user makes a format error while entering a terminal definition command or enters a command which is invalid for the terminal, the message

xxERR..

is displayed at the terminal.

TERMINAL DEFINITION PARAMETERS

The following descriptions explain the terminal definition parameters used in the terminal definition command format. Default values are not shown since they vary with the terminal class. (Refer to table F-1 for the default values.)

TERMINAL CLASS (TC)

The TC parameter specifies a value from 1 to 15 that associates the terminal type with a set of terminal characteristic parameters. Any terminal being used in the network belongs to one of the following 15 terminal classes.

TC	Terminal Type
1	M33, M35, M37, M38 Teletypes
2	CDC 713-10
3	Reserved
4	IBM 2741
5	M40
6	Hazeltine 2000
7	CDC 751-1, 752, 756
8	Tektronix 4014
9	HASP Protocol
10	200 User Terminal
11	CDC 214
12	CDC 711-10

TABLE F-1. TERMINAL CLASSES

Terminal Type		M33, M35, M37, M38	CDC 713-10	Reserved	IBM 2741
Parameter	Mnemonic				
Terminal Class	TC	1	2	3	4
Control Character	CT	ESC (Any) [†]	ESCAPE (Any) [†]		ATTN % (Any) [†]
Backspace Character	BS	CTRL/H (Any) [†]	← (Any) [†]		BACKSPACE (Any) [†]
Cancel Character	CN	CTRL/X (Any) [†]	CTRL/X (Any) [†]		ATTN ((Any) [†]
Abort Output Line Character	AL	CTRL/X (Any) [†]	CTRL/X (Any) [†]		ATTN ((Any) [†]
User Break 1 (Interruption Character)	B1	CTRL/P (Any) [†]	CTRL/P (Any) [†]		ATTN : (Any) [†]
User Break 2 (Termination Character)	B2	CTRL/T (Any) [†]	CTRL/T (Any) [†]		ATTN) (Any) [†]
Carriage Return Idle Count	CI	2 (0-99, CA)	0 (0-99, CA)		2 (0-99, CA)
Line Feed Idle Count	LI	1 (0-99, CA)	0 (0-99, CA)		1 (0-99, CA)
Page Width	PW	72 (0-255)	80 (0-255)		132 (0-255)
Page Length	PL	0 (0-255)	0 (0-255)		0 (0-255)
Page Wait	PG	N (Y,N)	N (Y,N)		N (Y,N)
Parity	PA	E (Z,O,E,N)	E (Z,O,E,N)		E (Z,O,E,N)
Special Editing	SE	N (Y,N)	N (Y,N)		N (Y,N)
Transparent Input Mode Delimiter	DL ^{††}	X 0D C2043 (X any C1-4096 TO)	X 0D C2043 (X any C1-4096 TO)		X 6D (X 6D)
Input Device	IN ^{††}	KB (KB,PT,XK,XP,X)	KB (KB,PT,XK,XP,X)		KB (KB,PT,XK,XP,X)
Output Device	OP	PR (PR,PT)	DI (DI,PT)		PR (PR,PT)
Echoplex Mode	EP	N (Y,N)	N (Y,N)		N/A

TABLE F-1. TERMINAL CLASSES (Contd)

Terminal Type		M40	Hazeltine 2000	CDC 751-1	Tektronix
Parameter	Mnemonic				
Terminal Class	TC	5	6	7	8
Control Character	CT	CTRL/P (Any) [†]	ESC (Any) [†]	ESC (Any) [†]	ESC (Any)
Backspace Character	BS	No default (Any) [†]	CTRL/H (Any) [†]	← (Any) [†]	CTRL/H (Any) [†]
Cancel Character	CN	CTRL/X (Any) [†]	CTRL/X (Any) [†]	CTRL/X (Any) [†]	CTRL/X (Any) [†]
Abort Output Line Character	AL	CTRL/X (Any) [†]	CTRL/X (Any) [†]	CTRL/X (Any) [†]	CTRL/X (Any) [†]
User Break 1 (Interruption Character)	B1	CTRL/F (Any) [†]	CTRL/P (Any) [†]	CTRL/P (Any) [†]	CTRL/P (Any) [†]
User Break 2 (Termination Character)	B2	CTRL/T (Any) [†]	CTRL/T (Any) [†]	CTRL/T (Any) [†]	CTRL/T (Any) [†]
Carriage Return Idle Count	CI	1 (0-99, CA)	0 (0-99, CA)	0 (0-99, CA)	0 (0-99, CA)
Line Feed Idle Count	LI	3 (0-99, CA)	3 (0-99, CA)	12 (0-99, CA)	0 (0-99, CA)
Page Width	PW	74 (0-255)	74 (0-255)	80 (0-255)	74 (0-255)
Page Length	PL	0 (0-255)	0 (0-255)	0 (0-255)	0 (0-255)
Page Wait	PG	N (Y,N)	N (Y,N)	N (Y,N)	N (Y,N)
Parity	PA	E (Z,O,E,N)	E (Z,O,E,N)	E (Z,O,E,N)	E (Z,O,E,N)
Special Editing	SE	N (Y,N)	N (Y,N)	N (Y,N)	N (Y,N)
Transparent Input Mode Delimiter	DL ^{††}	X 0D C2043 (X any C1-4096 TO)	X 0D C2043 (X any C1-4096 TO)	X 0D C2043 (X any C1-4096 TO)	X 0D C2043 (X any C1-4096 TO)
Input Device	IN ^{††}	KB (B,PT,XK,XP,X)	KB (KB,PT,XK,XP,X)	KB (KB,PT,XK,XP,X)	KB (KB,PT,XK,XP,X)
Output Device	OP	DI (DI,PT)	DI (DI,PT)	DI (DI,PT)	DI (DI,PT)
Echoplex Mode	EP	N (Y,N)	N (Y,N)	N (Y,N)	N (Y,N)

TABLE F-1. TERMINAL CLASSES (Contd)

Terminal Type		HASP Protocol	200UT	CDC 214	CDC 711-10
Parameter	Mnemonic				
Terminal Class	TC	9	10	11	12
Control Character	CT	% (Any) †	% (Any) †	% (Any) †	% (Any) †
Backspace Character	BS	N/A	N/A	N/A	N/A
Cancel Character	CN	((Any) †	((Any) †	((Any) †	((Any) †
Abort Output Line Character	AL	N/A	N/A	N/A	N/A
User Break 1 (Interruption Character)	B1	: (Any) †	: (Any) †	: (Any) †	: (Any) †
User Break 2 (Termination Character)	B2	) (Any) †	) (Any) †	) (Any) †	) (Any) †
Carriage Return Idle Count	CI	N/A	N/A	N/A	N/A
Line Feed Idle Count	LI	N/A	N/A	N/A	N/A
Page Width	PW	80 (0-255)	80 (0-255)	80 (0-255)	80 (0-255)
Page Length	PL	0 (0)	13 (0-255)	13 (0-255)	16 (0-255)
Page Wait	PG	N/A	Y (Y,N)	Y (Y,N)	Y (Y,N)
Parity	PA	N/A	0 (0)	0 (0)	0 (0)
Special Editing	SE	N/A	N/A	N/A	N/A
Transparent Input Mode Delimiter	DL ††	N/A	N/A	X03	X 03 (X 03)
Input Device	IN ††	KB (KB)	KB (KB)	KB (KB,X)	KB (KB,XK)
Output Device	OP	DI (DI)	DI (DI)	DI (DI)	DI (DI)
Echoplex Mode	EP	N/A	N/A	N/A	N/A

TABLE F-1. TERMINAL CLASSES (Contd)

Terminal Type		CDC 714	CDC 731-12 CDC 732-12	CDC 734
Parameter	Mnemonic			
Terminal Class	TC	13	14	15
Control Character	CT	% (Any) [†]	% (Any) [†]	% (Any) [†]
Backspace Character	BS	N/A	N/A	N/A
Cancel Character	CN	((Any) [†]	((Any) [†]	((Any) [†]
Abort Output Line Character	AL	N/A	N/A	N/A
User Break 1 (Interruption Character)	B1	: (Any) [†]	: (Any) [†]	: (Any) [†]
User Break 2 (Termination Character)	B2	) (Any) [†]	) (Any) [†]	) (Any) [†]
Carriage Return Idle Count	CI	N/A	N/A	N/A
Line Feed Idle Count	LI	N/A	N/A	N/A
Page Width	PW	80 (0-255)	80 (0-255)	80 (0-255)
Page Length	PL	16 (0-255)	13 (0-255)	13 (0-255)
Page Wait	PG	Y (Y,N)	Y (Y,N)	Y (Y,N)
Parity	PA	0 (0)	0 (0)	0 (0)
Special Editing	SE	N/A	N/A	N/A
Transparent Input Mode Delimiter	DL ^{††}	X 03 (X 03)	N/A	N/A
Input Device	IN ^{†††}	KB (KB,XK,X)	KB (KB)	KB (KB)
Output Device	OP	DI (DI)	DI (DI)	DI (DI)
Echoplex Mode	EP	N/A	N/A	N/A
[†] Any ASCII character except NUL, STX, EOT, LF, CR, DEL, =, and space. ^{††} DL is not a legal parameter in the TERMDEF request. ^{†††} XK, XP, and X are not legal values for the IN parameter in the TERMDEF command.				

TC	Terminal Type
----	---------------

13	CDC 714-10/20
14	CDC 731-12, 732-12
15	CDC 734

A terminal that is not shown as belonging to a terminal class may still be operational if it has been assigned to a terminal class having similar characteristics and whose parameters have been changed as necessary. Site personnel makes these changes to define correctly the operating characteristics of the terminal. The terminal class modified to support the new terminal type has operating parameter values different from the default values shown in table F-1.

Terminals connected to the network through auto-recognition lines are auto-recognized as belonging to terminal class 1, 4, 9, 10, or 13. When the terminal being used on an auto-recognition line is other than terminal class 1, 4, 9, 10, or 13, the TC parameter must be changed to identify correctly the terminal class.

CONTROL CHARACTER (CT)

The CT parameter establishes the character to be used to identify terminal definition commands. When this character is entered as the first character in a logical line (or transmission block), data following is assumed to be a terminal definition. Any character in the ASCII 128-character set except equals (=) can function as the control character. The character specified by the CT parameter is the character represented by control in the terminal definition command format.

BACKSPACE CHARACTER (BS)

This parameter specifies the character to be used to delete the previous input character. Any character in the ASCII 128-character set except equals (=) can function as the backspace character. It is possible to backspace only to the beginning of the current physical line; additional backspaces are discarded. When a page width of 0 is selected, the characters are sent to TAF in multiples of 150 characters regardless of the page width. Backspacing across these 150-character boundaries is not possible.

CANCEL CHARACTER (CN)

The CN parameter specifies the character to be used to cancel the logical line currently being input. Any character in the ASCII 128-character set except equals (=) can function as the cancel character. When the cancel character is entered as the last character before the Ⓢ is pressed, the entire logical line in progress is cancelled and *DEL* appears at the terminal. If part of the current logical line has already been transmitted to TAF, a flag is sent to inform TAF that the cancel character has been entered.

The cancel character cancels any auto-input characters that a task sends to the terminal.

ABORT OUTPUT LINE CHARACTER (AL)

The AL parameter specifies the character to be used to abort an output logical line. Any character in the ASCII 128-character set except equals (=) can function as the abort output line character. The current output line is discarded when the abort output line character is entered as the only character in a logical line (entering the abort output line character followed by Ⓢ).

USER BREAK 1 CHARACTER (B1)

The B1 parameter specifies the character that, when entered as the only character in a logical line (B1 character followed by Ⓢ), causes user break 1 indication to be set. A user break 1 indication is returned to the task if the task was performing a SEND with recall. A break 1 supervisory message is sent to ITASK, if the SEND was done without recall. Any character in the ASCII 128-character set except equals (=) can function as the user break 1 character.

USER BREAK 2 CHARACTER (B2)

The B2 parameter specifies the character that, when entered as the only character in a logical line (B2 character followed by Ⓢ) causes a user break 2 indication to be set. A user break 2 indication is returned to the task if the task was performing a SEND with recall. A break 2 supervisory message is sent to ITASK if the SEND was done without recall. Any character in the ASCII 128-character set except equals (=) can function as the termination character.

CARRIAGE RETURN IDLE COUNT (CI)

The CI parameter specifies the number of idle characters to be inserted into the output stream after a carriage return. The CI parameter can be assigned any value from 0 through 99, or CI=CA can be specified to restore the carriage return idle count to the default value shown in table F-1.

LINE FEED IDLE COUNT (LI)

The LI parameter specifies the number of idle characters to be inserted into the output stream following a line feed. The LI parameter can be assigned any value from 0 through 99, or LI=CA can be specified to restore the line feed idle count to the default value shown in table F-1.

PAGE WIDTH (PW)

The PW parameter establishes the maximum number of characters that a terminal can display on one output line. Page width can be set to any decimal value from 0 through 255. When output occurs at the terminal, the carriage is advanced to the beginning of the next physical line when the number of characters displayed equals the page width. When a page width of 0 is selected, the display is never advanced to a new physical line because of page width; PW=0 selects an infinitely long line.

NOTE

The PW parameter should only be used to establish a page width equal to the physical page width of the terminal being used. Setting page width to other values can cause unpredictable results.

PAGE LENGTH (PL)

The PL parameter establishes the maximum number of physical lines that can be displayed as one page. Page length can be set to any value from 0 through 255. PL=0 sets an infinitely long page. During output at the terminal, when the number of lines output is one less than the PL parameter selected, a page boundary is reached. If page waiting has been selected (refer to the PG parameter), output stops until an entry at the terminal (CR) is made to continue output of the next page.

PAGE WAIT (PG)

The PG parameter specifies whether page waiting is to be performed for the terminal. Page waiting is selected by entering PG=Y and cancelled by entering PG=N. When page waiting is selected, output stops at each page boundary, and terminal input (CR) is required before the next page is displayed. When the current output page is not full, and another page is available for output, the terminal displays

OVER..

at the end of displayed output. In character mode, a page is determined by the number of physical lines specified by the PL parameter. During transparent mode operation, the end of transparent output indicator (for example, the delimiter character) is interpreted as a page boundary. Page waiting is in effect only when the output device is defined to be a CRT display (OP=DI).

PARITY SELECTION (PA)

The PA parameter specifies the type of parity that the terminal generates on input and expects on output. Parity can be odd (O), even (E), zero (Z), or no (N). In even or odd parity, the network checks (or generates) the eighth bit of each character as a parity bit. The network clears the bit before transmitting the character to TAF. In zero parity, the network does not check for parity and automatically clears the eighth bit. If no parity is specified, the network transmits all 8 bits as data, without checking parity.

SPECIAL EDITING (SE)

The SE parameter specifies special editing mode for the terminal. Special editing mode is selected by entering SE=Y and deselected by entering SE=N. When the terminal is in special editing mode, the cancel input, backspace, and line feed characters are sent to TAF as part of input data.

TRANSPARENT INPUT MODE DELIMITER (DL)

The DL parameter specifies transparent input mode text delimiters. Three types of delimiters can be selected; characters, character count, and timeout. Each is optional, but at least one must be selected. A delimiter character can be any character and is specified with Xcc, where cc is the two-hexadecimal-digit representation of the delimiter character in the terminal's character set code. Character count delimiter is specified with Cvalue, where value can be any decimal value from 1 through 4096. The timeout delimiter is selected by specifying TO, which indicates a 200- to 400-millisecond timeout. Delimiters can be entered in any order; trailing commas can be deleted. Terminal classes 12 and 13 are configured with the ETX key as the transparent mode delimiter; that configuration cannot change. The TAF user normally need not be concerned with the DL parameter.

INPUT DEVICE (IN)

The IN parameter specifies the input device as a keyboard in character mode (KB), a keyboard in transparent mode (XK), a paper tape reader in character mode (PT), or a paper tape reader in transparent mode (XP). Paper tape input is allowed in keyboard mode, but X-ON characters are not sent to start the paper tape reader. For transparent mode, the transparent delimiter (DL) character must have been established before transparent mode is entered or the default delimiter for the terminal class is used. The TAF user is not normally concerned with this parameter.

OUTPUT DEVICE (OP)

The OP parameter specifies the output device as a printer (PR), CRT display (DI), or paper tape punch (PT). Printer and CRT display are functionally equivalent except for the page wait feature. The terminal user can punch a paper tape in any mode, but the proper X-OFF characters are provided only if PT is selected and the terminal is not in transparent mode. The TAF user is not normally concerned with this parameter.

ECHOPLEX MODE (EP)

The EP parameter selects input character echoing. If EP=Y is specified, each character received by the network is echoed back to the terminal just as it was received. This mode is effective only for terminals which have echoplex capability. For terminals having a HALF/FULL duplex switch, if this switch is in the HALF position when EP=Y is specified, each subsequent character entered is double printed (initially when it is typed and again when it is echoed back by the system). Placing the switch in the FULL position allows only the character echoed back to the terminal to be printed.

EP=N clears the EP=Y setting. Characters received by the network after EP=N is specified are not echoed back to the terminal. Characters entered at the terminal are not printed when the HALF/FULL duplex switch is in the FULL position. Only system-generated output appears at the

terminal. Placing the switch in the HALF position allows keyboard entry to be printed.

CODE SET (CD)

The CD parameter allows the user to change terminal character sets after the initial terminal identification

sequence. To do this, the user enters CD=A Ⓢ. The user has 60 seconds to change the existing terminal character set on the terminal (for example, change type balls). The user completes the character set recognition sequence by pressing)Ⓢ. Refer to Entering Transaction Subsystem in section 1 for more details.

Table G-1 shows the keys used by the supported terminal classes to advance the carriage or cursor to the beginning of the next line and/or transmit messages from the terminal. Appendix F describes the terminal classes. The keys listed in table G-1 are divided functionally into three types: carriage return, new line, and line feed. These keys are described by the functions they provide for the terminal user and how TAF interprets them.

Terminal user functions are:

Carriage return

This key terminates a message, advances the cursor or carriage to the beginning of the next line, and transmits the message. It also transmits any other messages which may have been stored in the terminal buffer. This key is the one normally referred to by the term CR in this manual (although the new line key can be used if available) because commands terminated by a carriage return obtain an immediate response.

New line

This key performs the same function as the carriage return but does not transmit the message. Rather, it causes the message to be stored in the terminal buffer until the carriage return key is used to transmit the entire buffer to TAF.

Line feed

This key advances the cursor or carriage to the beginning of the next line but does not terminate the message. Entering carriage return or new line causes the message to be terminated.

TAF interprets these keys as:

Carriage return

This key terminates a message.

New line

This key terminates a message.

Line feed

This key is transparent to TAF in all but one case because the network interprets this key to cause input line folding (entering one logical line on more than one physical line). This key is only recognized by TAF when the AP=S terminal definition command has been entered. (Refer to appendix F.)

TABLE G-1. TERMINAL KEY EQUIVALENCES

Terminal Class	Function		
	Carriage Return	New Line	Line Feed
1	RETURN	N/A	LINE FEED
2	RETURN	N/A	LINE FEED
3			
4	RETURN	N/A	ATTN
5	RETURN	N/A	NEW LINE
6	CR	N/A	LF
7	CARRIAGE RETURN	N/A	LINE FEED
8	RETURN	N/A	LF
9	varies	varies	varies [†]
10	SEND	RETURN	N/A
11	SEND	RETURN	N/A
12	ETX	NEW LINE	N/A
13	ETX	NEW LINE	N/A
14	ETX	NEW LINE	N/A
15	SEND	NEW LINE	N/A

[†]Different terminals operating under HASP protocol use different keys for this purpose.

TAF prevents situations from occurring in which the execution of a task is blocked because it is impossible under the circumstances to obtain the needed central memory (CM) space for the task, whether for an initial load or a memory increase. Several terms will be defined to aid in the description of how potentially blocked tasks are detected:

Minimum size of the total task area:

The total task area is the CM in which TAF will schedule tasks. It includes the space occupied by CM resident tasks. The area extends from the first word address into which tasks can be loaded to the maximum field length (MFL) of TAF. The size of this area fluctuates if the TAF/CRM data manager is used. The minimum size of this area occurs when the TAF/CRM buffer takes on its maximum size.

Minimum size of the transient task area:

The transient task area is that portion of the total task area not occupied by CM resident tasks. The minimum size of this area is defined to be the minimum size of the total task area minus the sum of the MFL's of all CM resident tasks.

One or more tasks will be considered to be potentially blocked under the following circumstances:

1. If the initial field lengths of the CM resident tasks are so large that they all could not be loaded together.

2. If the sum of the MFL's for the CM resident tasks exceeds the minimum size of the total task area.
3. If the MFL of a non-CM resident task exceeds the minimum size of the transient task area.

The following are the times when potentially blocked tasks are detected and the actions which are taken:

1. At TAF initialization. Potentially blocked tasks result in TAF being aborted.
2. When TAF is informed, via the LIBTASK TT option, of the update of a task library attached by TAF and the library contains potentially blocked tasks, the update will be rejected.
3. When an attempt is made to reduce the MFL of TAF via the K.MAXFL command to a value which results in potentially blocked tasks, the command will be rejected.

The above checking assumes that the maximum field length of TAF is set (by default, via K.MFL, or via K.MAXFL commands) to a value which can be attained.

INDEX

ABH (refer to Application block header)

Accessing transaction subsystem 1-4

ACT 2-10

Application block header

Building 2-12

Examining 2-13

Modifying 2-12

Application character type 2-10

Auto-input mode 1-3; 2-12

Batch

Interface 1-3; 5-1

Subsystem interaction 5-1

BEGIN request 2-28

BLDABH request 2-12

Blocked tasks H-1

BTRAN request 5-1

Build ABH 2-12

BWAITINP request 2-26

CALLRTN 2-4

Call tasks 2-3,4,5

CALLTSK request 2-5

CDCS 1-1,2; 3-1; 4-3

CEASE request 2-29

Chain 2-3,4,5

Change ACT 2-10

Change CMDUMP request values 6-1

Change I/O limit 2-26

Change task field length 2-3; 9-3

Character sets A-1

Character type 2-10

Characteristics of terminal (refer to Terminal characteristics)

CMDUMP request 6-3

Communication block

Defined 2-1

Dump 6-5

Load overflow data 2-9

Release overflow data 2-24

TAF-originated transactions 11-3,4

Create task library 9-4

Create xxJ file 4-4

CYBER Database Control System 1-1,2; 3-1; 4-3

CYBER Record Manager data manager 1-1,2; 4-1,2

Data base

Activate 4-6

Describe 8-1

Journal files 4-4

Maximum number of 4-6

Data manager

Concurrency 1-1; 3-1

Requests 2-3

xxJ files 4-1,2,3,4

Dead locks 3-1

Define terminal characteristics 2-15

Delete tasks from task library 9-5

Describe users and data bases 8-1

Determine if user is logged in 2-31

Determine subsystem for task 2-27

Diagnostics B-1

DIAL directive 8-1

DMS statement 4-6

DSDUMP request 6-1

Dump

Communication block 6-5

Data buffers 6-3,5

Exchange package 6-5

TAF field length 6-5

Task field length 6-3,5

ECS 1-6

End task 2-29

Entering transaction subsystem 1-4

Entry header, journal file 4-4

Error messages B-1

Examine ABH 2-13

Execute named task 10-2

Extended core storage 1-6

Extend field length 2-3; 9-3

Extract bit string from word 2-31

EXTRACT request 2-31

File

Definitions D-1

Descriptions D-1

Files

Journal 4-4

JOUR0 4-4

Network 8-1

Pool 4-1

Procedure 4-1

Reserved D-2

System 4-1

TCF 4-1,6; D-1

Trace 4-1

xxERPF 4-1

xxJ 3-1,2,3,4

xxJORN 4-4

xxTASKL 9-1

xxTFIL 4-1

Format effectors 2-12,13,14

GETABH request 2-13

Global task dump limit 6-5

GTDL 6-5

Ignore task in replacement file 9-5

IIO request 2-26

Implement application 1-5

Initial task 1-3; 10-1; 11-1

Initiate new transaction chain 2-5,6

Input/output

Limit 2-26

Requests 2-7

Input overflow 2-9,24

Insert bit string into word 2-32

INSERT request 2-31
Installing TAF E-1
Interactive virtual terminal 1-3; 2-15
Interrogate TST 2-24
ITASK 1-3; 10-1; 11-1
ITL request 2-29
IVT 1-3; 2-15

JOUR0 file 4-4
Journal files 4-4
JOURNL request 4-5

KDIS 10-1
K display commands 2-30; 6-1,4
K.DSDUMP command 6-4
K.DUMP command 6-5
K.DUMPLIM command 6-5
KPOINT request 2-30
KTSDMP utility/request 6-5

Libraries 9-1
LIBTASK utility/control statement 9-1
Line feed and transmission keys G-1
LOADCB request 2-9
Load overflow blocks 2-9
LOGIN request 2-31
Log in terminal to TAF 1-4
Log out task 10-1
Log out terminal from TAF 2-30; 10-1
LOGT request 2-29
LOGT system task 10-1

Macros Inside front cover; 11-1
Manipulate user area of TST 2-25
Memory
 Changing from tasks 2-3; 9-3
 Dump requests 2-3; 6-1
 Dumps 6-1
 Requests 2-3
Message abort 10-1
Modify ABH 2-12
Modify terminal characteristics 2-15
Move data from terminal to task 2-26
MSABT 10-1
Multimainframe 1-6

NAM 1-3; 11-2,3
NCT 11-5
NDL 1-6
Network
 Access Method 1-3; 11-2,3
 Communication table 11-5
 Definition language 1-4
 File origination 8-2
 Files 8-1
 Interface 1-3
NEWTRAN request 2-6

OFFTASK 10-1
On-line transaction processing 1-1
Operator interface 7-2

Procedure file 4-1,4; 7-1

RA+1 calls 1-6; 2-26
Read data from terminal 2-8
RECALAL request 2-28
Record data on journal file 4-4
Recovery 7-1
Release overflow blocks 2-24
RELSCB request 2-24
Requests
 Data manager 2-3
 Input/output 2-7
 Journal file 2-3; 4-4
 Memory 2-3
 Memory dump 2-3; 6-1
 System 2-3
 TAF executive Inside front cover
 Task
 Control 2-28
 Scheduling 2-3
 Utility 2-31
Reserved files D-2
RESTRICT clause processing 3-1

Sample deck structures C-1
Schedule tasks 2-3; 9-4
Send message to user 2-16
SEND request 2-16
Sense switches 7-1
SETCHT request 2-10
Set GTDL 6-5
Set time limit for task 2-29
Simultaneously logged in terminal 1-5
Specify communication block location 2-28
Start transaction using batch job 5-1
Status of transaction 2-6
STRAN macro 11-1
Subcontrol points 1-6
Submit batch job using task 5-3
SUBMT request 5-3
Subtransaction codes 11-1
Supervisory messages 2-10
Suspend processing 2-30
SYSMSG 10-1
System
 Files 4-1
 Journal files 4-4
 Requests 2-3
 Task library 9-1
 Tasks 10-1

Tables
 Network communication (NCT) 11-5
 Terminal status (TST) 8-2

TAF
 Configuration file 4-6; D-1
 Control point 1-1,2
 Data manager 4-1
 Executive requests Inside front cover; 1-3
 Installation E-1
 Originated transactions 11-2,3
 Procedure file 4-1,4; 7-1
TARO request 2-25
Task
 Chain 2-4,5,6
 Control requests 2-28
 Defined 1-3; 2-1

- Dump 6-1
- Field length changes 2-3; 9-3
- Library
 - Create 9-4
 - Defined 9-1
 - Delete task from 9-5
 - Ignore task in replacement file 9-5
 - Update 9-5
- Pause 2-28,30
- Queuing 9-1,4
- Residency 9-1
- Scheduling priority 9-4
- Scheduling requests 2-3
- Type 9-1
- Utility requests 2-31
- TASKLIB 9-1
- Tasks
 - Blocked H-1
 - Call 2-3,4,5
 - End 2-29
 - Schedule 2-3; 9-4
 - Time-originating 11-1
- TCF file 4-1,6; D-1
- TERMDEF request 2-15
- Terminal
 - Characteristics
 - Define 2-15; F-1
 - General 1-5; 11-3,4,5
 - Modify 2-15; F-1
 - Definition commands F-1
 - Network description file 8-1
 - Status request 11-5
 - Status table
 - Figure of 8-2
 - Interrogate 2-24
 - Manipulate user area 2-25
- TIMCNT macro 11-1
- Time-originating tasks 11-1
- Total data manager 4-1,3
- Total task area H-1
- TPSTAT request 2-27
- TRAN macro 11-1
- TRANCHK request 2-6
- TRANS 11-1
- Transaction
 - Codes 11-1

- Defined 1-1
- Processing 1-3
- Subsystem
 - Entering 1-4
 - Interaction 5-1
 - TAF-originated 11-2,3
- Transient task area H-1
- Transparent mode 2-12
- TRANT 11-1
- TSIM request 2-24
- TST (refer to Terminal status table)
- TTOT 11-1
- Type ahead 11-5

- Updating task library 9-5

- VALNET statement 8-1
- VALNET validation 8-1
- Validate terminal network description file 8-1

- Wait for terminal input 2-8
- WAIT request 2-30
- WAITINP request 2-8

- XTASK 10-1,2
- xxERPF file 4-1; D-1
- xxJ files
 - CDCS data manager 4-3
 - Creating 4-4
 - CRM data manager 4-2
 - No data manager 4-3
 - TAF data manager 4-1
 - Total 4-3
- xxJORN file 4-4
- xxTASKL file 9-1
- xxTFIL file 4-1

COMMENT SHEET

MANUAL TITLE: CDC Transaction Facility Version 1 Reference Manual

PUBLICATION NO.: 60455340

REVISION: D

NAME: _____

COMPANY: _____

STREET ADDRESS: _____

CITY: _____ STATE: _____ ZIP CODE: _____

This form is not intended to be used as an order blank. Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

CUT ALONG LINE

AA3419 REV. 4/79 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

TAPLE

STAPLE

FOLD

FOLD



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS

PERMIT NO. 8241

MINNEAPOLIS, MINN.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION

Publications and Graphics Division

ARH219

4201 North Lexington Avenue

Saint Paul, Minnesota 55112



CUT ALONG LINE

FOLD

FOLD

TITLE: CDC Transaction Facility Version 1 Reference Manual

PUBLICATION NO.: 60455340

REVISION: D

External A/D

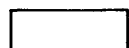
CONTROL DATA CORPORATION

Publications and Graphics Division
4201 North Lexington Avenue
St. Paul, Minnesota 55112

REASON FOR CHANGE:

DATE 12-05-80

Revised to reflect NOS 1.4 at PSR level 530 and to make technical and editorial corrections. Support of TAF/CDCS interface is included. This edition obsoletes all previous editions.



CORPORATE HEADQUARTERS, P.O. BOX 0, MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.



CONTROL DATA CORPORATION